HORS-SÉRIE

compatibles PC magazine

PROGRAMMATION & LISTINGS

MAITRISER SON IMPRIMANTE

Configuration
Traducteur résident
Compatibilité
Gestion

BASIC

Fichiers en accès direct
Gestion bancaire
Générateur de tableaux
Agenda

PASCAL

Dessins
Gestion imprimante

dBASE

Interrogation de fichiers

GRATUIT
DISQUETTE AB-CLUB
6 LOGICIELS

HORS SÉRIE N°1 JUIN 1989
BELGIQUE 160 FB SUISSE 6,50 FS CANADA 5,95 $ CAN

Photo de couverture: Didier Crété.
Illustrations Marc Houplain

Disquette gratuite avec 6 logiciels, offerte par AB Club
Bon à découper en page 62

Les listings de ce numéro hors-série sont disponibles sur disquette: bon de commande en page 62.

compatibles PC magazine

SOMMAIRE

N°1 HS - JUIN 1989

MAITRISER SON IMPRIMANTE

Votre imprimante a bon caractère 3
Configuration des imprimantes 9
Rendre compatible son imprimante 14

BASIC

Les fichiers en accès direct 21
Gestion bancaire 44
Un générateur de tableaux 45
Agenda 63

PASCAL

Dessiner en Turbo Pascal 32
Gestion d'imprimante en T.P 36
Un menu dans un fichier BAT 40
Envoi de caractères de contrôle à une imprimante 42

dBASE

Interrogation dBase 57
Un browse dans une fenêtre 26

PC magazine **est édité par Laser Presse SA,** 5-7, rue de l'Amiral Courbet, 94160 Saint-Mandé.
REDACTION.
Directeur de la rédaction : Jacques Eltabet. **Secrétaire de rédaction** : Marie Peretti. **Ont également collaboré à ce numéro :** M.Archambault, J.P. Pruniaux, Roland Lopez, Didier Huet, Louala Nasser, Stéphane Perroy, Claude Raymond, Eric Servagnat ,Pierre Squelart.
FABRICATION. Maquette : Thierry Martinez, Laurence Floquet, Alain Lamargo.
ADMINISTRATION.
Diffusion : Bertrand Desroches.
Abonnements : au (1) 43 98 01 71. **Comptabilité** : Sylvie Brouxel.
PUBLICITE.
Régie publicitaire : Néo-Média, 5-7, rue de l'Amiral Courbet 94160 Saint-Mandé. Tél.:(1) 43 98 22 22. Télécopie: (1) 43 28 72 12. **Directeur de la publicité** : Thierry Cagnion. **Assistant du directeur de la publicité** : Marc Rosenzweig.
Directeur de la publication : Jean Kaminsky. **Impression** : La Haye-les-Mureaux. **Commission paritaire** : 68966. **Dépôt légal** : 2e trimestre 1989.

VOTRE IMPRIMANTE A BON CARACTERE

Avec l'arrivée des compatibles PC, les codes ASCII sont passés de 128 à 256. Les imprimantes disponibles avant cette nouveauté n'en tiennent pas compte. Voici la recette pour les remettre à jour.

S'il est un standard reconnu et appliqué dans le monde entier, c'est bien celui-là : tout caractère utilisé en informatique, qu'il soit imprimable ou non (les caractères de contrôle), est placé dans un tableau, sa place étant fixée par une norme internationale: c'est le code ASCII. Dès le départ de la micro informatique, ce code a été conçu pour être représenté sur 7 bits, donnant un code de 0 à 127.
De 0 à 31 (décimal), on trouve les caractères de contrôle non imprimables qui permettent de transmettre des ordres au dispositif récepteur (écran, imprimante, fichier, etc.). Le plus célèbre est le code 13 qui commande le retour à la ligne (*carriage return*), suivi du code 12 qui est le saut de ligne (*form feed*).
De 48 à 57 sont rangés les chiffres de 0 à 9, puis de 65 à 90 les lettres majuscules de A à Z, et enfin de 97 à 122 les lettres minuscules.
Les trous dans la numérotation donnée ci-dessus sont occupés par les caractères spéciaux tels que virgule, point, et autres signes mathématiques. Cette répartition, qui peut sembler en désordre, correspond en fait à une croissance ordonnée de la valeur des bits de la représentation numérique.

Cette table de création américaine n'a pas tenu compte des caractères avec accents des autres langues, et de toute façon la place manquait avec 128 possibilités au départ. L'arrivée du PC d'IBM a popularisé l'extension de la norme ASCII sur huit bits soit 256 caractères. Les 128 premiers ne sont pas modifiés. En revanche, à partir du code 128 on trouve les caractères avec accents des langues européennes, quelques caractères grecs et signes mathématiques, et des caractères semi-graphiques autorisant la confection de tableaux par exemple, avec des traits continus, donc de présentation plus agréable que l'utilisation classique des - + ! et autres symboles.

Il est évident que si les écrans sont prévus dès lors pour représenter ces caractères additionnels à la norme de départ, il n'en est pas de même pour les imprimantes acquises antérieurement. Vu la longévité de celles-ci, et surtout leur coût, faut-il les mettre à la poubelle ou se résoudre à imprimer des textes démunis des tableaux bien dessinés et de nos chers accents ? Il faut reconnaître qu'ils posent bien des problèmes d'orientation, vive les correcteurs orthographiques !

Une solution : Printer Adapter 3

Une solution existe, à condition que l'imprimante en votre possession soit capable de travailler en mode graphique : la plus populaire est l'imprimante Epson type MX-80, équipée pour le plus ancien modèle du dispositif Graftrax. Cette solution consiste à remplacer le caractère absent par son équivalent en mode graphique. Nous avons trouvé dans le commerce un utilitaire qui permet d'adapter pratiquement toutes les imprimantes existantes et futures à un PC, il s'agit de *Printer Adapter 3* diffusé par Logiciels et Médias. Comme son nom ne l'indique pas, c'est un produit français écrit par Michel Bujardet.
La disquette est accompagnée d'une notice de dix-huit pages expliquant de façon très

succincte son utilisation. L'ensemble comprend les fichiers suivants:
— INSTALLE.BAT, utilisable par les possesseurs d'un disque dur, procède à la création d'un répertoire appelé PRAD (PRinter ADapter) et y effectue la copie de la disquette.
— PRAD.COM est le programme de création ou de modification des pilotes d'imprimante, pour adapter votre imprimante si les pilotes fournis ne vous conviennent pas. Nous allons revenir en détail sur son utilisation.
— PRAD.DEF est un fichier de configuration dont l'usage est réservé à Printer Adapter.
— PRAD.COM constitue le 'moule' des futures adaptations.
— PRAD—xxx sont les programmes d'adaptation fournis pour résoudre un certain nombre de cas:
• PRAD—UNI permet d'utiliser une imprimante quelconque en composant les caractères accentués par superposition de deux caractères (à condition que l'imprimante accepte le retour arrière), et utilise une représentation approchée des caractères semi-graphiques de la norme IBM, par utilisation des caractères classiques + - I. Cette solution est évidemment celle du pauvre mais le principal est de pouvoir imprimer dans l'attente du cadeau du père Noël.
• PRAD—EPS permet de piloter une imprimante type Epson graphique sans ROM IBM. Les caractères accentués sont composés également par la superposition de deux caractères, par exemple pour obtenir é on superpose e et ', les caractères semi-graphiques sont reproduits, comme sur une imprimante compatible IBM, par dessin graphique de chacun de ceux-ci. Cela vous évite donc de solliciter votre ministre des finances pour un nouvel investissement.
• PRAD—APL permet de piloter dans les mêmes conditions, les imprimantes Apple, DMP,ImageWriter 1 ou 2.
• PRADUTI.EXE est un utilitaire permettant de configurer votre imprimante, en fonction de ses possibilités, pour différents styles d'impression, d'utiliser l'imprimante comme d'une machine à écrire, dans ce cas chaque ligne est imprimée au fur et à mesure, et surtout d'obtenir la liste des pilotes d'imprimantes présents en mémoire avec leurs ports d'affectation, plus la possibilité de les rendre actifs ou non (dans le cas d'impression de graphique).
MSPool est fourni en prime, puisque c'est un programme du domaine public, pour continuer à utiliser votre compatible pendant l'impression. Une documentation en français est fournie sur la disquette.

Mode d'emploi

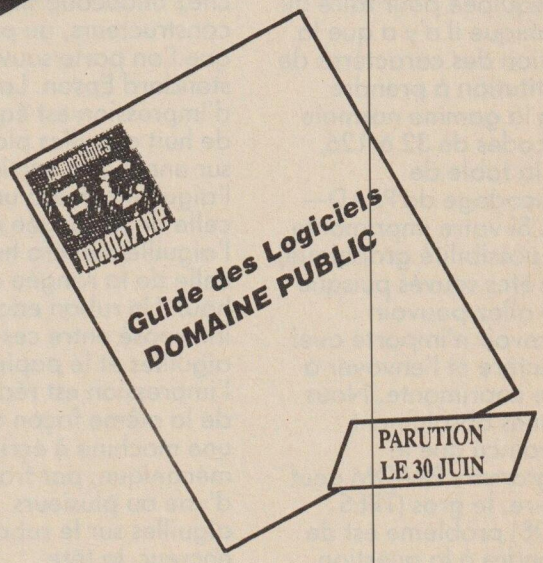
Vous choisissez l'un des trois programmes évoqués ci-dessus (PRAD—UNI, PRAD—EPS, PRAD—APL), et vous le placez sur la disquette de démarrage (ou dans la racine de votre disque dur). Vous pouvez lancer son exécution manuellement, ou incorporer son nom dans votre fichier AUTOEXEC.BAT. Ce programme est du type résident, il est chargé en mémoire, se met en attente, permet donc d'exécuter un programme quelconque (traitement de texte par exemple). A partir de là, chaque fois que vous envoyez un caractère à l'imprimante, au lieu d'être adressé directement sur le port de sortie, il est détourné de son chemin habituel pour passer par le programme PRAD—xxx qui est alors réactivé d'une façon totalement transparente à l'utilisateur pour l'occasion.
Le travail de PRAD—xxx consiste à consulter une table dite de "transcodage" comprenant deux colonnes. La colonne de gauche contient les valeurs numériques des codes ASCII (de 0 à 255), un code par ligne. Pour chacune de ces lignes, on trouvera dans la colonne de droite le, ou les, codes à envoyer réellement à l'imprimante. Si le code à envoyer correspond aux possibilités d'impression de votre imprimante, sa valeur est envoyée sans aucune transformation au port de sortie, le temps perdu est alors négligeable. En revanche, dans le cas contraire, c'est la chaîne de caractères placée à droite de la table de transcodage qui est envoyée, et suivant son mode de traitement, le temps d'impression sera plus long. Vu les résultats obtenus, cela vaut bien un petit sacrifice.
Pour éviter les conflits entre les différents programmes résidents, un ordre de chargement est conseillé sur la notice:
MSPool
PRAD—xxx
PRADUTI R
rem autres programmes
D'autre part, il ne faut pas charger deux fois le même programme résident, car il y a risque de blocage du système. C'est pour éviter ce problème qu'il est conseillé d'incorporer les ordres de chargement dans le fichier AUTOEXEC.BAT, vous disposerez alors de votre vieille imprimante comme d'une petite jeunesse.

Personnalisation de la table de transcodage

Nous avons décrit plus haut le mode de fonctionnement de la table de transcodage, en mentionnant en particulier la chaîne de caractères placée dans la colonne de droite, chaîne réellement envoyée à l'imprimante. L'adaptation de cette table aux caractéristiques de l'imprimante, consiste à modifier le contenu de cette colonne, pour chacune des lignes des codes ASCII des caractères à reproduire. Ce travail est possible grâce à l'utilitaire PRADM.COM. Pour modifier une table, il est préférable de lui attribuer un autre nom, ou plus exactement une autre seconde partie de nom. En effet, le nom est composé de trois parties. Les deux premières parties correspondent au nom principal respectant les règles de MS-DOS, entre autres : 8 caractères maximum. Ce nom commence toujours par PRAD— et il est suivi par trois lettres qui différencient les versions de tables de transcodage. La troisième partie est toujours .COM. Avant toute modification vous devez donc produire un nouveau fichier sur lequel vous allez travailler, par exemple avec la commande:
COPY PRAD—EPS.COM PRAD—ESS.*
ESS étant la contraction de ESSAI, il est évident que vous pouvez modifier directement un fichier existant, dans ce cas il n'est pas nécessaire d'en effectuer une copie, mais souvenez-vous que l'original sera perdu.

Dans le cas de notre exemple, sa modification sera lancée par la commande: PRADM ESS , PRADM est l'utilitaire de modification, et ESS est la deuxième partie du nom du fichier à modifier. Un choix d'options est proposé: Changer la table de transcodage, Choisir le port de sortie sur lequel l'imprimante est connectée, Mettre à zéro la table de transcodage (Pour recommencer entièrement celle-ci), Quitter le programme , Aide; donnant un très (très) court résumé de la documentation.
Cette possibilité est en fait l'atout principal de *Printer Adapter*. La table de transcodage est affichée à l'écran par page de 22 caractères ASCII. On passe d'une page à l'autre au moyen des touches 'PgUp' pour remonter vers la première, et la touche 'PgDn' pour redescendre vers la dernière.
Une page est composée de quatre éléments:
— une bande supérieure donne l'intervalle des caractères présentés dans la page,
— une bande inférieure rappelle les actions principales possibles: touche F2 pour quitter le mode modification avec sauvegarde de la table de transcodage active, touche DEL pour annuler le caractère pointé par le curseur, touche F10 pour obtenir à l'écran le caractère du code ASCII valeur zéro (ne pas confondre avec le code ASCII du caractère zéro), qu'il n'est pas possible d'afficher autrement, sa représentation correspond à Control-@ (nous y reviendrons).
— La zone centrale est divisée en deux tableaux, celui de gauche affiche sur chacune des 22 lignes la valeur numérique du code ASCII, et sa représentation à l'écran entre parenthèses, un code non imprimable est représenté avec deux caractères, en premier (accent circonflexe) suivi de @ pour le code 0, A pour le code 1, B pour le code 2, etc... jusqu'au caractère — (souligné) pour la valeur 31. Pour vous y retrouver vous remarquerez un décalage de 64 avec les caractères "normaux".
— Le tableau de droite est le plus intéressant pour nous, puisqu'il affiche le ou les caractères envoyés réellement à l'imprimante pour chacun des codes ASCII de 0 à 255 (maximum 61 caractères par code ASCII). La convention suivante est adoptée pour la représentation des codes ASCII non affichables (ou caractères de contrôle), ils apparaissent en surbrillance, par exemple le code 'Retour Chariot' (CR=Carriage return) valeur ASCII 13 ou encore Ctrl-M est représente par M en surbrillance (si votre moniteur le permet). Les touches 'flèches' du clavier permettent de déplacer le curseur clignotant sous l'une quelconque des positions de cette seconde partie du tableau de transcodage.
En parcourant la table en cours de modification vous constaterez que les caractères sans problèmes, c'est-à-dire ceux acceptés par votre imprimante, ne subissent pas de transformation,

NUMÉROS HORS SÉRIES



DES GUIDES INDISPENSABLES

par exemple on trouve ligne 65 à gauche la lettre A et à droite la même lettre A. Si votre imprimante accepte le retour arrière (Ctrl-H ou BS ou Backspace), le caractère é (e accent aigu) code ASCII 130 est obtenu en envoyant successivement les caractères donnés ci-après entre parenthèses (e Ctrl-H '), les parenthèses ne sont pas envoyées, le travail effectué par l'imprimante est celui-ci: impression du caractère e retour sur celui-ci et superposition du caractère ' ce qui donne au total é.
Il est évident que si vous disposez d'une imprimante moderne, toute cette gymnastique est inutile puisqu'il y a de forte chance qu'elle soit équipée avec la ROM compatible IBM, donc capable d'afficher directement les codes ASCII de 0 à 255. Pour les autres (moins fortunés ou plus conservateurs), reste à résoudre le problème de l'impression des caractères semi-graphiques ou exotiques divers.
Si votre imprimante n'est pas équipée pour faire du graphique il n'y a que la solution des caractères de substitution à prendre dans la gamme normale des codes de 32 à 126, voir la table de transcodage de PRAD—UNI. Si votre imprimante a la possibilité graphique, vous êtes sauvés puisque vous allez pouvoir concevoir n'importe quel caractère et l'envoyer à votre imprimante. Nous sommes absolument convaincu que le programme PRADM peut le faire, le gros (TRES GROS) problème est de répondre à la question stupide: comment?

Au début de notre article nous avons dit que la notice était succincte, en voici la preuve, puisqu'elle est absolument muette sur la façon de procéder.

Où l'on pousse la réflexion plus loin...

Normalement notre article devrait s'arrêter ici, mais nous avons poussé notre réflexion un peu plus loin, et nous vous donnons à la suite la façon de procéder avec en prime, la source d'un programme en Turbo-Pascal vous permettant de créer en interactif (ou presque), n'importe quel caractère dans une grille à définir jusqu'à un maxi de 16x16.

Mode graphique

Nous prenons comme modèle l'imprimante Epson MX-80, dont le mode de fonctionnement graphique se retrouve, quelquefois avec des modifications mineures, chez beaucoup de constructeurs, au point que l'on parle souvent de standard Epson. La tête d'impression est équipée de huit aiguilles placées sur une même verticale : l'aiguille numéro un est celle de la rangée du bas, l'aiguille numéro huit est celle de la rangée du haut. Un ruban encreur est interposé entre ces aiguilles et le papier : l'impression est réalisée de la même façon qu'avec une machine à écrire mécanique, par frappe d'une ou plusieurs aiguilles sur le ruban encreur, la tête d'impression se déplaçant longitudinalement sur la largeur du papier soit en 480 positions, ce qui correspond pour une ligne de 80 caractères à 480/80=6 pas par caractère, ou 960/80=12 pas.
Pour réaliser la frappe d'une aiguille, le constructeur a associé un octet de donnée à la tête d'impression, pas par hasard, puisqu'un octet est composé de huit bits, chacun de ces bits est associé à une aiguille. Le premier à l'aiguille numéro un, ainsi de suite. Si un bit a la valeur zéro, l'aiguille correspondante est immobile, et s'il a la valeur un, son aiguille vient frapper le ruban et retourne à sa place, cette action ayant laissé la trace d'un point sur le papier. Chaque bit représente un 'poids' suivant sa position, pour le calcul de la valeur de l'octet, suivant le schéma ci-après.

N° Aiguille	valeur
8	128
7	64
6	32
5	16
4	8
3	4
2	2
1	1

Pour activer les aiguilles 2, 4 et 7, par exemple, il suffit d'affecter à l'octet de contrôle la valeur calculée comme suit:

```
bit--) 1  2  3  4  5  6  7  8
       0 + 2 + 0 + 8 + 0 + 0 + 64 + 0 = 74 (-- Octet
```

Toutes ces opérations se passent à l'intérieur de l'imprimante, l'utilisateur dispose d'une convention, le standard graphique Epson, pour commander ce processus. Il suffit d'envoyer à l'imprimante le code ASCII 27 (Escape ou échappement) suivi d'une séquence de codes. Voyons en détail cette séquence:
ESC K b h ...

ESC est la valeur 27 (en décimal) du code Escape
K valeur ASCII 75 pour définir le mode graphique 480 points à remplacer par L (valeur 76) en mode 960 points
b représente le poids faible du nombre de positions de la tête d'impression compté sur 2 octets
h est le poids fort (nous y revenons après)
... est la suite des valeurs de l'octet de commande de la tête d'impression, une valeur par position de celle-ci.

Le b et le h ci-dessus nécessitent une petite explication: l'imprimante étant connectée au port parallèle de l'ordinateur, les données sont transmises sur huit fils parallèles. Vous savez que huit fils correspondent à un octet, d'où une plage de variation de valeur de 0 à 255. Comme nous devons pouvoir envoyer à l'imprimante jusqu'à 960 octets pour réaliser une ligne entière en caractères graphiques, la convention suivante est adoptée: en divisant le nombre par 256, on obtient un quotient et un reste. Par exemple 452 nous donne un quotient de 1 et un reste de 196. Le poids fort est de 1 et le poids faible est de 196, ce qui donne suivant la convention ci-dessus: b=196 et h= 1. Autrement dit, en Basic par exemple, le début de l'instruction se présenterait comme suit:

```
LPRINT
CHR$(27);K;CHR$(196);
CHR(1); [suivi des 452 octets
de la ligne graphique]
```

Pour résoudre notre problème, c'est plus simple puisque notre caractère sera défini en principe avec 7 octets d'où b=7 et h=0. Pour bien comprendre le mécanisme, prenons un exemple. Soit à imprimer

le caractère inscrit dans une grille de 7 colonnes et de 8 lignes donné ci-après:

```
 XXXXX
XX   XX
X X X X
X  X  X
X  X  X
X X X X
XX   XX
 XXXXX
```

Avec beaucoup d'imagination cela ressemble à un sablier. Pour composer notre ligne de commande à l'imprimante, nous commençons par analyser chaque colonne. La colonne 1 a toutes ses lignes occupées par des croix sauf les deux extrêmes, donc sa valeur est de:

0 +2 +4 +8 +16 +32 +64 +0 =126

La colonne 2 est calculée comme suit:

1 +2 +0 +0 +0 +0 +64 +128 =195

De la même façon, on trouvera les valeurs suivantes:
colonne 3 = 165
colonne 4 = 153
colonne 5 = 165
colonne 6 = 195
colonne 7 = 126
Nous pouvons maintenant construire notre chaîne à envoyer à l'imprimante, qui comprend la suite des nombres décimaux:
27 75 7 0 126 195 165 153 165 195 126

En Basic, nous écrivons l'instruction suivante:

```
120 LPRINT CHR$(27);K;
CHR$(7); CHR$(0);
CHR$(126);CHR$(195);
CHR$(165);CHR$(153);
CHR$(165);CHR$(195);
 CHR$(126)
```

Pour revenir à notre problème de personnalisation de notre table de transcodage, si nous décidons d'affecter ce caractère "sablier" au caractère de code ASCII 251 par exemple, nous faisons défiler les pages de 22 caractères jusqu'à ce qu'il soit possible de positionner le curseur sur la ligne 251, nous effaçons les caractères de la partie droite, que nous remplaçons par la frappe des 11 caractères dont les valeurs décimales sont données ci-dessus. Nous vous rappelons encore une fois, qu'il suffit pour entrer chaque valeur, d'appuyer sur la touche ALT et, sans la relâcher, de composer la valeur décimale. Ce qui doit vous donner une ligne composée comme suit:

[KG@~|N̄ÖN̄|~

Les caractères [G @ doivent être affichés à l'écran en surbrillance comme indiqué plus avant. Nous avons essayé de montrer la façon de procéder pour imprimer un caractère quelconque. Ce n'est pas compliqué, il suffit de beaucoup de méthode et de patience. Le plus fastidieux est de dessiner un caractère sur une grille et de calculer les valeurs des octets de chacune des colonnes. Pour vous aider, nous vous présentons un programme en Turbo-Pascal qui effectue ce travail.

Matricage d'un caractère

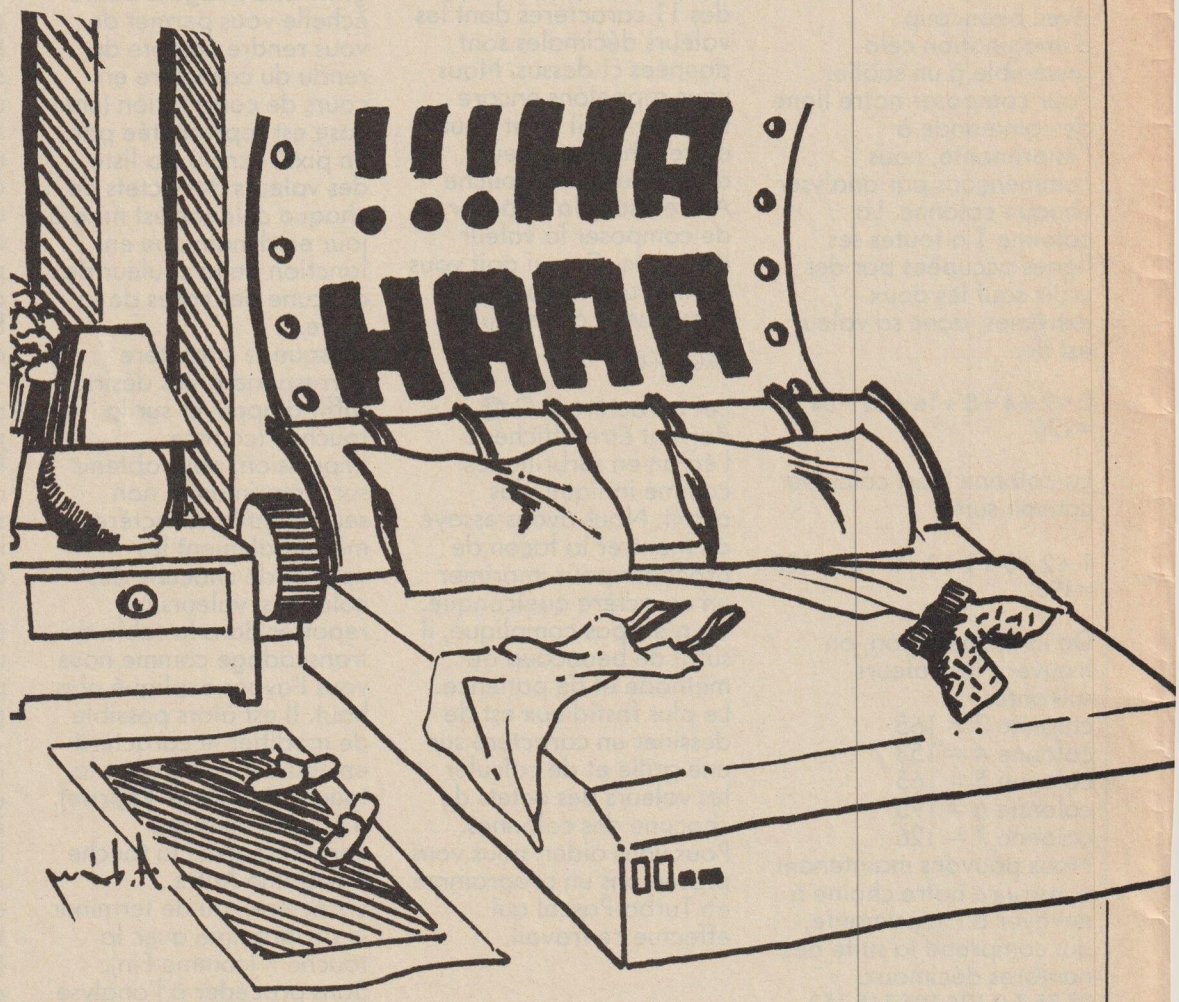
Au lancement du programme, vous choisissez la grille d'implantation du caractère à dessiner, jusqu'à un maximum de 16x16. La grille est affichée à grande échelle sur l'écran, et sur la partie droite de celui-ci, vous avez la liste des valeurs des octets de chacune des colonnes. Un curseur est dessiné dans la case inférieure gauche de la grille, ce curseur est déplacé à l'intérieur de la grille à l'aide des touches flèches du clavier. L'appui sur la touche RETOUR inverse la couleur de la case où se trouve le curseur. De noir, elle passe à blanc et vice-versa.
Sur le côté droit de la grille une image à petite échelle vous permet de vous rendre compte du rendu du caractère en cours de constitution (une case est représentée par un pixel écran). La liste des valeurs des octets de chaque colonne est mise à jour en dynamique en fonction de la couleur de chacune des cases de la grille.
Lorsque le caractère correspond à vos désirs, il suffit d'appuyer sur la touche I (comme Impression) pour obtenir sur l'imprimante, non seulement le caractère, mais également les valeurs de chacune des colonnes, valeurs à reporter dans la table de transcodage comme nous vous l'avons expliqué plus haut. Il est alors possible de modifier le caractère en cours d'étude avec la touche R (comme Reprise), d'étudier un autre caractère avec la touche A (comme Autre caractère), ou de terminer le programme avec la touche F (comme Fin).
Sans procéder à l'analyse détaillée de ce programme nous allons signaler quelques points particuliers.

Programme de matricage

Ce programme utilise les possibilités graphiques pour représenter la grille et le curseur. Le problème est qu'il existe principalement deux standards pour les compatibles: la carte CGA dont nous utilisons la définition monochrome 640x200, et la compatibilité Hercules monochrome de 720x348 (nous ne parlons pas du standard EGA inaccessible actuellement pour les amateurs).
L'autre problème avec Turbo-Pascal est l'obligation d'utiliser le ToolBox graphique pour commander la carte Hercules, ce qui est une solution très lourde pour un programme aussi simple. D'autre part, il n'existe pas de possibilité de compilation conditionnelle permettant de concevoir un programme fonctionnant dans les deux cas de figures. Les principes adoptés sont les suivants:
— pour le graphique en mode CGA, pas de problème puisque Turbo-Pascal (version 3.x) comprend les procédures standard Hires pour l'initialisation en mode 640x200 monochrome, Plot pour dessiner un point, Draw pour dessiner une ligne et TextMode pour désactiver le mode graphique ;
— pour le graphique en mode Hercules, nous utilisons les primitives incluses dans la carte via l'interruption 10h. En effet, Hercules (ou ses compatibles) fournit un logiciel utilitaire appelé INT10.COM, qui doit être appelé à l'allumage de la machine (par incorporation dans le fichier AUTOEXEC.BAT par exemple) après le traditionnel HGC FULL, dont le but est d'apporter une extension aux paramètres de l'interruption 10h. En particulier, nous utilisons l'attribut 40h pour initialiser le mode graphique en 720x348, l'attribut 42h pour effacer

l'écran graphique, l'attribut 41h pour restituer l'écran alphanumérique, l'attribut 44h pour donner la "couleur" du tracé, l'attribut 46h pour dessiner un point, les attributs 48h pour déplacer le curseur graphique "plume levée" et 49h pour le déplacer "plume baissée", et enfin l'attribut 4Bh pour écrire un texte en mode graphique (dans une matrice de 9x14 pixels. Comme vous pouvez le constater, sur la source Turbo-Pascal l'appel aux interruptions est très facile. Pour obtenir un programme identique dans les deux cas de figure, nous avons adopté la convention suivante: toutes les primitives graphiques et les paramètres propres à un mode graphique sont déportés dans un fichier appelé GENECAR.GRA qui sera incorporé dans notre programme au moment de la compilation par la directive {$I genecar.gra}.
Les procédures apportées par ce fichier sont: G—Ini pour l'initialisation des paramètres de dimension d'écran (G—Lecran et G—Hecran), de position de grille et ses pas en x et en y, et l'appel de la fonction d'initialisation propre à une carte. G—Ligne pour dessiner une ligne entre les deux couples de coordonnées fournis. G—Point pour dessiner un point. G—Write pour écrire un texte en mode graphique à un emplacement défini. G—Texte pour revenir en mode Texte. Nous préparons deux fichiers, GENECAR.CGA qui fera appel aux procédures standards graphique de Turbo-Pascal, et GENECAR.HGC qui fera appel aux paramètres de interruption 10h apportés par la carte Hercules. Avant la compilation, il suffit de choisir l'installation visée, et d'effectuer la copie du fichier Genecar.CGA ou Genecar.HGC dans le fichier Genecar.GRA (copie réalisée par exemple avec la commande DOS 'COPY GENECAR.HGC *.GRA'). Nous avons solutionné plusieurs problèmes au cours de cet article qui, nous l'espérons, n'a pas été trop indigeste : la transformation d'une imprimante quelconque en une imprimante compatible IBM en utilisant le logiciel du commerce *Printer Adapter 3* de Logiciels et Medias, l'utilisation du mode graphique d'une imprimante type Epson, et l'utilisation de la carte graphique Hercules à partir du Turbo-Pascal (nous reviendrons plus en détail sur ce sujet). Vu la longueur du texte, il vous faudra attendre le n°9 du *PC Magazine* pour avoir les listings.
Printer Adapter 3 : 1 097,05 F ttc.
Logiciels et Médias, 125, rue Saussure, 75017 Paris.

Pierre Squelart



PETITES ANNONCES GRATUITES

Pour acheter, revendre, ou échanger du matériel, pour établir des contacts, rédigez votre petite annonce très lisiblement, en majuscules

Nom : ..... *Prénom :* .....
Adresse : .....
C.P. : ..... *Ville :* ..... *Tél. :* .....
.....
.....
.....
.....
.....

Bon à renvoyer à :
Compatibles PC Magazine
5/7, rue de l'Amiral Courbet
94160 Saint-Mandé

CONFIGURATION DES IMPRIMANTES

«Je ne peux imprimer les minuscules accentuées... Au lieu des caractères graphiques j'ai des lettres en italiques... Ce logiciel me fait des sauts de lignes et de pages en toute fantaisie... »

Qui accuser ? Le PC, l'imprimante, le vendeur, ou le logiciel ? Neuf fois sur dix, le fautif, c'est l'utilisateur : vous ! Mais attention, vous avez beaucoup de circonstances atténuantes... En effet, configurer correctement une imprimante est loin d'être une chose simple, et il faut reconnaître que les notices, même en français, sont partiellement rédigées en "chinois". Pour se tirer d'affaire tout seul il est absolument indispensable d'assimiler certaines notions générales.

Les règles de base ne sont pas ardues mais assez... "spéciales". En fin d'article, nous évoquerons le cas des imprimantes laser. Le problème est généralement complexe parce que le concepteur d'une imprimante veut que son produit convienne à tous les micro-ordinateurs, et ce des antiques Apple II jusqu'au Compaq 386, en passant par les Atari, Amstrad CPC, etc. Pas simple ! Aussi prévoit-il toute une batterie de micro-interrupteurs destinés à se plier aux caprices de ceux qui ont conçu ces micro-ordinateurs ô combien différents.

Principe de fonctionnement d'une imprimante

Par son câble d'entrée, l'imprimante ne reçoit que des nombres, à savoir les "codes ASCII" des caractères à tracer sur papier, mais aussi des "codes Epson" qui sont des "consignes" d'écriture.
La partie électronique rappelle beaucoup celle d'un ordinateur : microprocesseur, mémoire vive RAM, mémoire fixe ROM. En final, il ne s'agit pas de balayage vidéo, mais de commandes purement électriques : un moteur pour faire avancer le papier, un autre pour déplacer la "tête d'impression", et dans celle-ci des électro-aimants miniatures pour actionner les neuf (ou vingt-quatre) "aiguilles" qui inscriront de minuscules points noirs sur le papier.

Parlons d'abord du contenu de la **ROM** :
— Un "programme résident" qui traduit le code ASCII reçu par le câble — exemple : 68 pour un D majuscule —, par une suite de commandes aux aiguilles associées au déplacement latéral de la tête. Une telle précision de micro-mécanique nous laisse rêveur, surtout à 240 caractères par seconde !
— Cette ROM comprend aussi l'ensemble des codes Epson qui constituent le "langage résident" de la machine. Ils vont paramétrer l'impression, par exemple le changement de taille des caractères.
Pour la **RAM**, disons qu'il y a deux mémoires vives différentes :
— le "buffer des consignes" contenant les codes Epson actuellement en vigueur, par exemple écrire en gros caractères et en gras ;
— le "buffer d'impression" qui, lui, est constamment renouvelé puisque l'on y stocke les codes ASCII des caractères à imprimer. Selon le prix de la machine, ce buffer va de quelques dizaines d'octets à 16, 32 ou 64 ko.
Ceci nous amène à une évidence : la mécanique de l'imprimante ne peut pas suivre le débit en caractères de votre PC. Celui-ci est régulé par l'imprimante qui lui dit «stop ! mon buffer est plein pour l'instant... ça y est, je suis de nouveau prête ("Ready") à en recevoir». Il y a donc un dialogue entre les deux machines, et ce par l'intermédiaire de certains fils du câble de liaison parallèle.

Les codes de consignes

Parmi les 255 valeurs numériques possibles qu'elle reçoit par le câble, voici comment l'imprimante identifie les codes de consignes des caractères à imprimer :
— Est à imprimer tout code compris entre 32 et 255, (NDLR : sauf le 127, le vieux caractère DEL, son origine date des rubans perforés où pour effacer un caractère déjà perforé un seul moyen : perforer toute la colonne, 127 en binaire).
— A mettre dans le buffer de consignes les codes entre 0 et 31. Si code=27 (ESC) mettre également les un ou deux codes qui suivent. Exemples de consignes :
- code 15 --> écriture en condensé.
- codes 27 puis 69 (lettre E) --> écriture en gras. (ESC E)
- codes 27, 120 (lettre x), 49 (chiffre 1) --> qualité courrier (ou NLQ ou roman). C'est le code Epson ESC x 1

Ces exemples font partie du "codage Epson", qui est devenu la norme internationale. Mais il existe aussi d'autres codages, heureusement fort peu usités :
— les **codes IBM** commencent tous par Esc (27). De ce fait, les codes 0 à 31 sont des caractères graphiques supplémentaires. Reconnaissons que l'idée était bonne, mais elle venue bien trop tard pour supplanter le standard Epson.
— les **codes maisons** spécifiques à tels modèles d'imprimantes, exemples des anciens modèles de Seikosha, Amstrad (DMP 1), Tandy... Cette pratique semble enfin abandonnée.

Les tables de caractères

La bonne vieille norme des codes ASCII, partiellement respectée par les IBM PC, est en gros la suivante :
— Les codes sur sept bits (donc inférieurs à 128) rassemblent tous les symboles usuels, à savoir signes de ponctuation, chiffres, lettres majuscules et minuscules.
— Les codes sur huit bits (128 à 255) reprennent ces mêmes caractères mais en italique, plus quelques signes graphiques.
— Les douze codes 35, 36, 64, 91, 92, 93, 94, 95, 123, 124, 125 et 126 sont réservés à la représentation des lettres spécifiques à treize pays, dont la France pour ses minuscules accentuées. Ces changements

de "nationalités" s'effectuent par micro-interrupteurs ou par le code Epson ESC R n.
Voyons ce qu'est devenue cette norme, revue et corrigée par IBM pour les PC depuis 1981 :
— Les codes sur sept bits de 32 à 127 restent inchangés.
— Les sept codes "internationaux" sont figés en pays = USA.
— Les codes sur huit bits sont des caractères graphiques parmi lesquels on trouve les minuscules accentuées françaises, espagnoles, norvégiennes, etc.
— Les codes 1 à 31 «peuvent être» des symboles graphiques, sauf si l'on utilise les "codes Epson" pour le codage des consignes.
Donc deux conclusions :
● **Notre imprimante doit posséder les "caractères graphiques IBM"** (généralement le jeu n°2). Or certains modèles anciens, ou «dédiés» à d'autres micros, ne possèdent pas cette «EPROM IBM».
● **L'imprimante doit être impérativement "switchée USA"** et surtout pas French; le code Epson ESC R ne sert plus.

Les "tables IBM"

Le manuel de votre imprimante présente plusieurs tables de caractères, indiquant ce qu'imprimeront les codes de 0 à 255. On y trouve la table "antique et standard" avec les italiques, à ne pas prendre, et quatre "tables IBM" :
On les reconnaît par la présence, entre 128 et 255, de nombreux symboles en traits simples et doubles (pour dessiner des encadrements). Ce sont les mêmes que vous pouvez obtenir à l'écran.
Certaines tables proposent des symboles intéressants de codes 0 à 31. Hélas ! pas question de les choisir (du moins pour l'instant), car ils sont incompatibles avec la majorité des logiciels. Un exemple convaincant est le joli symbole du code 12 : sur écran, c'est l'effacement (CLS), et sur imprimante c'est le saut de page ! (FF).
Autre exemple irritant pour qui désire faire une recopie d'écran avec les quatre caractères triangles 16, 17B, 30 et 31. On choisira donc la table qui déclare les codes 0 à 31 (1F en hexa) comme étant des consignes (BS, LF, FF, CR...), et tout le reste en caractères imprimables.
Généralement, c'est la table "IBM 2", mais cette nomenclature mal normalisée peut être différente dans certains manuels d'imprimantes. Voici un exemple extrême : le manuel de l'Epson FX-800 propose quatre "tables Epson" plus quatre "tables IBM". Eh bien, celle qui correspond à la description que nous venons de donner s'appelle ici "table Epson n° 4" !
Afin de vérifier que votre configuration est correcte, lancez le petit programme Basic "CARACTPC", et conservez précieusement la page qu'il imprimera.
Nota : Les symboles graphiques pour encadrements existent en traits simples et doubles, or certaines imprimantes anciennes ou bas de gamme traduisent en traits simples les caractères en traits doubles (Brother M-1509, Epson FX-85...).

Rôle des micro-interrupteurs

Ils sont destinés à fixer des paramètres par défaut lors de la remise sous tension de l'imprimante. Cette simple phrase implique donc que :
— la manœuvre d'un micro-interrupteur sous tension n'a aucun effet immédiat, elle est même dangereuse pour l'électronique ;
— les codes Epson (consignes) envoyés par le PC seront prioritaires sur ces "consignes par défaut", qui peuvent donc être contredites ou annulées. Une seule exception, la forme du chiffre zéro (barré ou non) pour laquelle il n'existe pas de code Epson correspondant (à ma connaissance).
Certains choix sont obtenus par la combinaison des positions de deux ou trois micro-interrupteurs. Exemple ON-ON-OFF, OFF-ON-ON...
Il arrive aussi qu'un bloc de micro-interrupteurs ait des fonctions complètement différentes selon la position d'autres micro-interrupteurs ! C'est le cas de modèles Citizen.
Selon le modèle et la marque de l'imprimante, le nombre de ces micro-interrupteurs varie entre 8 et 32. Il y a donc une anarchie totale pour la nature et le nombre des "questions" posées. Nous allons éluder cela dans le paragraphe suivant.

Positionnement des «micro-interrupteurs» de l'imprimante

Précaution numéro 1 : notez sur une feuille les positions d'origine puis celles que vous aurez établies. Feuille à fixer dans le manuel.
Précaution numéro 2 : éteindre l'imprimante...
Forme du zéro : barré, sauf si vous faites surtout du traitement de texte. Bien que pour ce dernier il soit fréquent d'utiliser O majuscule pour zéro.
Taille des caractères (Size) : normale (ou pica) = 10 cpi (caractères par pouce) = 80 par ligne.
Police (Font) : normale ("draft"). Les polices NLQ (= ROMAN = qualité courrier) ou "sanserif" seront établies par les touches du panneau frontal ou par codes Epson.
Espacement proportionnel (PS) : non, surtout pas, sauf par option dans un traitement de texte.
Nationalité : USA.
Codes Epson/IBM : Epson.
Italiques/Graphiques : Graphiques (tables IBM)
Table IBM : le plus souvent 2. A vérifier dans le manuel.

```
10 ' CARACTPC - Caractères imprimés par PC - M.A. - 01/89
20 LPRINT CHR$(27);"@";CHR$(27);"E"
30 LPRINT TAB(10);"Correspondances Codes ASCII/caractères pour Compatibles
PRINT:LPRINT
40 FOR A= 31 TO 254 STEP 10
50 FOR I=A TO A+9
60 IF I=31 OR I=127 THEN LPRINT USING"######";I;:LPRINT SPC(2);:GOTO 90
70 IF I>255 THEN 90
80 LPRINT USING"######";I;:LPRINT" ";CHR$(I);
90 NEXT
100 LPRINT:LPRINT
110 NEXT
120 LPRINT CHR$(27);"@";CHR$(12);
130 END
```

TRUCS ET ASTUCES

Input buffer (RAM caractères à imprimer) : oui ou taille maxi. L'imprimante "rend la main" au PC avant d'avoir terminé l'impression. Seul cas où ce buffer est gênant : la mise au point des programmes d'édition.
LF (LINE FEED, saut de ligne supplémentaire) : non avec les PC et la majorité des micros actuels.
FL (FORM LENGTH, longueur de page 11 ou 12 pouces) : pour le papier continu, regardez la boîte du papier listing et choisissez en conséquence.
Skip Over : non. C'est un saut de 1 pouce (25,4 mm) en bas de chaque page avec papier en continu. Parfois utile si ces listings sont ensuite photocopiés ou mis en classeur.
Feuille à feuille/Continu : selon votre alimentation en papier. Attention, cela n'assure pas la commutation mécanique de l'entraînement du papier (friction ou traction par picots).
Feuille à feuille manuel/automatique (bac à feuilles) : le plus souvent, c'est un ergot du bac qui établit automatiquement cette commutation.
PE (*paper end detector*, détection de fin de papier) : oui, sauf en cas d'alimentation manuelle en feuilles séparées ou enveloppes. Souvent, le positionnement en "feuille à feuille" neutralise de lui-même le PE. A vérifier.
Buzzer : oui, c'est plus prudent.
SLCT IN : oui (ON). En OFF, la machine attend que le micro la débloque par la broche n°36. Or celle-ci n'est pas toujours soudée, dans le câble du côté PC...
Interface série : certaines imprimantes ont en plus un connecteur d'entrée "série RS 232 c" à 25 broches. En ce cas, on trouve des micro-interrupteurs pour paramétrer le signal en accord parfait avec ce qui a été défini (par soft) sur le micro : par exemple, 9 600 bauds, 8 bits, 1 bit d'arrêt, pas de parité. Cela a peu d'intérêt avec les PC, car la liaison "parallèle" type Centronics est bien préférable !
Nota : ces micro-interrupteurs miniatures sont mécaniquement fragiles, ils ont été conçus pour être positionnés une fois pour toutes, ne les surmenez pas... Assurez-vous que leurs leviers sont bien en butée car, contrairement aux interrupteurs classiques, ils ne renferment pas de ressorts internes...

Problèmes de sauts de pages

Prenons le cas le plus banal, papier continu avec longueur 11 pouces, et bien sûr le micro-interrupteur en question mis sur "11 pouces".
Comme l'interligne normal est de six lignes au pouce, la machine sait qu'une page fait 11 x 6 = 66 lignes, qu'elle va comptabiliser au cours de l'impression. Mais comment mettre le compteur de lignes à zéro en début de page ? Deux méthodes :
a - la mise sous tension.
b - le code Epson ESC (codes 27 et 64), qui a aussi pour effet de vider le buffer des consignes.
C'est surtout ce dernier rôle de ESC que l'on connaît, d'où un usage souvent abusif en programmation pour supprimer des consignes d'écriture, alors qu'il existe toujours un code d'annulation pour chacune d'elles. On comprend alors le pourquoi de sauts de pages soit disant "fantaisistes".
Cas du saut de page anticipé par le code 12, exemple en Basic LPRINT CHR$(12) : ce dernier ne fait pas avancer le papier de 11 pouces comme on le croit, mais du complément de lignes restant à faire pour terminer la page, d'où reprise de l'écriture en haut de la page suivante. Et c'est là qu'apparaît un autre piège qui provoque un décalage d'une ligne à chaque page : il fallait ajouter un point-virgule, LPRINT CHR$(12);
Autre cas, vous disposez d'une imprimante à chariot large avec bac à feuilles garni en "format commercial" (A3). Vous décidez d'élargir encore votre tableau et pour cela d'**imprimer "à l'Italienne"**, c'est-à-dire en orientant les feuilles dans le sens de la longueur. Bonne idée, mais pensez alors à prévenir votre imprimante que maintenant les pages sont moins hautes...

Problèmes de sauts de lignes

Un saut de ligne est commandé par le code 13 (retour à gauche de la tête d'impression), suivi du code 10 (avancement papier d'une ligne). Ainsi, pour LPRINT "A" (code 65), l'imprimante reçoit en fait les codes 65, 13 et 10.
Beaucoup se sont cru lésés en constatant qu'en écriture en condensé il n'y avait pas 132 caractères par ligne mais toujours 80 ! Cela vient des PC qui, par défaut, insèrent un saut de ligne après le quatre-vingtième caractère. C'est la taille du "buffer d'imprimante". On peut très facilement modifier cela :
— sous DOS faire MODE LPT1:255,6 (longueur 255, 6 lignes au pouce),
ou bien
— en Basic faire WIDTH "LPT1:",255.
Lorsque la tête de l'imprimante arrivera en bout de course, il y aura insertion automatique d'un saut de ligne.
On résume
Le saut de ligne est provoqué par :
— la fin d'impression d'une ligne, exemple LPRINT "A" ;
— la taille maxi du buffer imprimante ;
— la "butée mécanique" du chariot.
A partir de cela, on va comprendre certains sauts de lignes farceurs :
1 - FOR N=1 TO 8:LPRINT A$(N):NEXT
Si la longueur de A$(3) est égale à la taille du buffer ou à la largeur maxi du chariot, il y aura une ligne blanche au-dessous...
2 - Vous avez une imprimante largeur normale (8 pouces soit 20,3 cm), en écriture condensée, vous avez fixé le buffer d'imprimante à 255. L'édition est correcte. Votre logiciel tourne à présent sur le même papier mais sur une imprimante à chariot large, et la tête imprime ses fins de lignes sur le rouleau caoutchouc... Il fallait fixer le buffer à 132.
3 - Vous êtes de nouveau en pica (80 caractères/ligne) mais le buffer est resté fixé à 132, et vous imprimez une longue chaîne. Sur la seconde ligne saut de ligne après 52 caractères. Normal, 80 + 52 = 132 ; fin du buffer... Il est prudent de fixer le buffer à un multiple de la longueur ligne.

Problèmes de compatibilité

Acte 1
Pour concurrencer les imprimantes X, le fabricant de la marque Y doit répondre à deux impératifs :
— être compatible avec les vieux modèles X, d'où respect de certaines normes telles que les "codes Epson", la fiche d'entrée type "CENTRONICS", etc.
— proposer des nouveautés dont X ne disposait pas. On invente ainsi la "qualité courrier", la "double hauteur", l'espacement proportionnel... Oui, mais alors, si c'est nouveau ce n'est plus "standard" !
Acte 2
Et ceux qui fabriquent des logiciels ? Ils ont intérêt à ce que leur (cher) produit puisse lui aussi s'adapter à ces imprimantes en évolution permanente. Ce qui sous-entend qu'il doit savoir tirer profit d'un modèle moderne haut de gamme (proposer des "plus"), sinon l'acheteur de l'imprimante aurait le droit de se sentir frustré par ce logiciel.
Aussi présente-t-on dans "INSTALL" un choix possible sur une multitude d'imprimantes. Ces mini programmes s'appellent des «drivers d'imprimantes». Mais, bien souvent, la vôtre n'y figure pas... Si c'est le cas, essayez "Epson RX" (l'ancêtre du standard actuel) ou "Epson FX", c'est du passe-partout.
Pour que votre programme soit "portable", évitez d'utiliser les codes Epson peu ou mal normalisés, le pire étant ESC ! n qui permet facilement de modifier la

taille et la forme des caractères. Ainsi les ESC ! de l'Epson FX 80 ne ressemblent absolument pas à ceux de l'Epson FX 800.
Alors quelle compatibilité peut-on espérer entre les ESC ! d'une NEC et ceux d'une Star ?

Le cas des imprimantes laser

● On ne peut utiliser que des feuilles coupées. Pas de listings en continu ni de liasses auto-carbonées.
● Puisque la machine prépare la totalité de la page avant impression, celle-ci demandera une durée identique, qu'il y ait un seul mot ou cinquante lignes en petits caractères. On ne parle plus de débits en caractères par seconde (cps) mais en pages par minute.
● Mémoriser le graphisme d'une page entière implique beaucoup de mémoire RAM, 512 ko minimum en mode texte. Pour le graphisme c'est "nettement" insuffisant (75x75 points au pouce carré). Pour un graphisme aussi fin que sur une imprimante à aiguilles, il faut un méga-octet. Pour être plus fin, il faut 1,5 à 2 méga-octets (300x300 points au pouce carré). Et la RAM reste toujours aussi chère...
● Pour imprimer à l'italienne, on ne touche pas au bac de papier, on demande que l'écriture laser se fasse dans l'autre sens. Ce sont les modes "Portrait" et "Landscape" (Paysage).
● Une imprimante laser est très silencieuse. Elle débite plus qu'un modèle rapide à aiguilles. La qualité de l'impression est très supérieure (traits plus noirs avec bords plus nets).
● Pas question d'envisager une machine à plusieurs couleurs, le prix serait monstrueux.
Il y a des dizaines de fabricants d'imprimantes laser, mais vous remarquerez entre elles un certain «air de famille» car on y retrouve les mêmes «cœurs» (ensemble optique + mécanique). Il s'agit principalement des cœurs Canon et Ricoh. Avantage pour nous, il y aura moins de modèles de "toners" qu'il existe de modèles de rubans d'imprimantes !

Problèmes spécifiques aux imprimantes laser

Actuellement, il s'agit essentiellement de carence de polices de caractères ou de "driver" approprié dans tel logiciel. A part quelques rares modèles (Memorex) qui possèdent une "émulation Epson FX" d'origine et sans bugs, l'obtention de caractères autres que le Roman taille normale constitue soit un énorme problème financier — *cf* les polices en EPROM enfichables —, soit du suspens avec les "polices téléchargeables" en RAM.

Correspondances Codes ASCII/caractères pour Compatibles PC

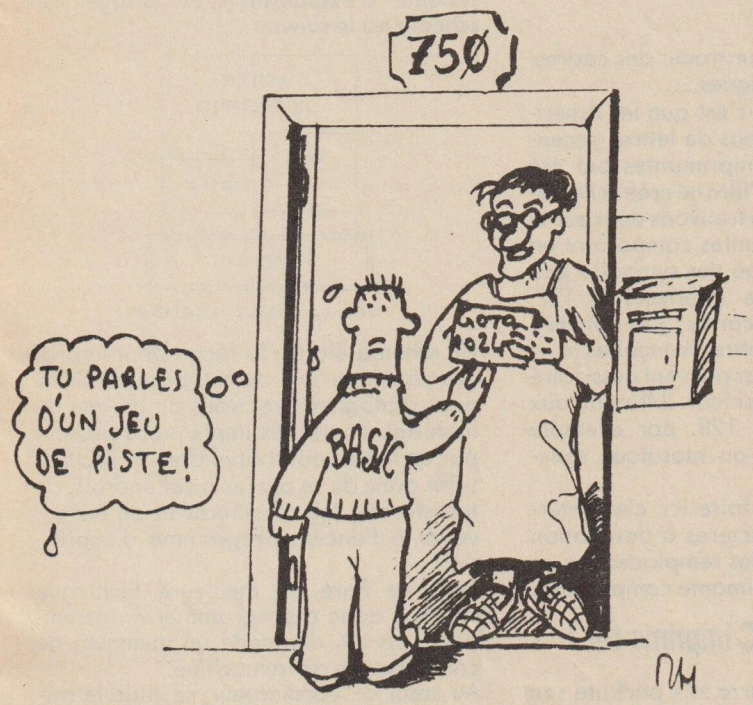
31		32		33	!	34	"	35	#	36	$	37	%	38	&	39	'	40	(
41	)	42	*	43	+	44	,	45	-	46	.	47	/	48	0	49	1	50	2
51	3	52	4	53	5	54	6	55	7	56	8	57	9	58	:	59	;	60	<
61	=	62	>	63	?	64	@	65	A	66	B	67	C	68	D	69	E	70	F
71	G	72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O	80	P
81	Q	82	R	83	S	84	T	85	U	86	V	87	W	88	X	89	Y	90	Z
91	[	92	\	93	]	94	^	95	_	96	`	97	a	98	b	99	c	100	d
101	e	102	f	103	g	104	h	105	i	106	j	107	k	108	l	109	m	110	n
111	o	112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w	120	x
121	y	122	z	123	{	124	¦	125	}	126	~	127		128	Ç	129	ü	130	é
131	â	132	ä	133	à	134	å	135	ç	136	ê	137	ë	138	è	139	ï	140	î
141	ì	142	Ä	143	Å	144	É	145	æ	146	Æ	147	ô	148	ö	149	ò	150	û
151	ù	152	ÿ	153	Ö	154	Ü	155	¢	156	£	157	¥	158	₧	159	ƒ	160	á
161	í	162	ó	163	ú	164	ñ	165	Ñ	166	ª	167	º	168	¿	169	⌐	170	¬
171	½	172	¼	173	¡	174	«	175	»	176	░	177	▒	178	▓	179	│	180	┤
181	╡	182	╢	183	╖	184	╕	185	╣	186	║	187	╗	188	╝	189	╜	190	╛
191	┐	192	└	193	┴	194	┬	195	├	196	─	197	┼	198	╞	199	╟	200	╚
201	╔	202	╩	203	╦	204	╠	205	═	206	╬	207	╧	208	╨	209	╤	210	╥
211	╙	212	╘	213	╒	214	╓	215	╫	216	╪	217	┘	218	┌	219	█	220	▄
221	▌	222	▐	223	▀	224	α	225	β	226	Γ	227	π	228	Σ	229	σ	230	µ
231	τ	232	Φ	233	Θ	234	Ω	235	δ	236	∞	237	φ	238	∈	239	∩	240	≡
241	±	242	≥	243	≤	244	⌠	245	⌡	246	÷	247	≈	248	°	249	∙	250	·
251	√	252	ⁿ	253	²	254	■	255											

Nota : par "émulation Epson FX", nous entendons la compatibilité totale avec une imprimante matricielle équipée d'un bac à feuilles ; à savoir toutes les tailles et formes de caractères et les consignes en codes Epson. Il suffit alors de débrancher le câble de la matricielle et de le brancher sur la laser.
La plupart des imprimantes laser ne se configurent pas au moyen de micro-interrupteurs mais de menus déroulants sur un petit écran LCD. Les choix sont conservés en RAM par une batterie interne.
Les "précurseurs", tels Hewlett-Packard avec ses "Laser Jet", tentent d'imposer un standard nouveau en ce qui concerne les codes de consignes (très complexe !) et le nom de polices nouvelles. Elles sont très belles et alléchantes mais gare à la facture de ces options quasi obligatoires.
Autre ennui classique, la plupart de ces machines se réservent des marges de sécurité à gauche et droite de la feuille. Cela n'apparaît pas en traitement de texte, à cause des marges fixées par le logiciel, mais une copie d'écran aura ses bords amputés de quelques caractères.

Et les imprimantes à jet d'encre ?

Super silencieuses, ultra rapides, très belle qualité d'écriture, elles se configurent comme une matricielle et leurs prix deviennent enfin abordables.
Il serait donc logique qu'elles supplantent à terme les modèles à aiguilles, oui mais...
Cette technologie a connu beaucoup de problèmes : buses qui se bouchent, encres qui bavent ou qui sèchent trop lentement, nécessité d'un papier spécial. On commence seulement à voir un modèle enfin fiable et sans histoires, mais avec des lacunes et des bugs dans la ROM. Alors attendons encore...

M. Archambault



CTI VOUS offre...

le **meilleur** choix de logiciels pour PC et Compatibles au prix **incroyable** de

200 F les 4 disquettes

A1 : AS EASY AS : aussi facile que... Un tableur très puissant, avec macros, graphismes intégrés et imprimables depuis votre feuille de calcul. (256 col. 1024 lignes).

A2 : WAMPUM : Un gestionnaire de fichier avec 2 modes de fonctionnement : utilisateur et développeur. Entièrement compatible DBASE III. (Config. 512 K. Disque dur).

A3 : ANTI-VIRUS : Une disquette contenant tous les programmes de détection existant à ce jour (Flu-shot, HDSentry, Scan, DProtect, etc...).

A4 - A5 : LEÇONS LOTUS : 2 disques pour vous apprendre à manipuler lotus, 500 écrans couleurs et graphiques, avec des exemples, des macros, des applications...

A6 : PC-WRITE : Le standard des traitements de texte, avec touche d'aide permanente, déplacement rapide dans le texte, marqueur, copieur, fenêtrage d'écran, etc...

A8 : MONOPOLY : Est-il bien utile de vous décrire ce jeu ? C.G.A. nécessaire.

A7 : LQ : Vous produirez avec cette disquette un texte de haute qualité sur une imprimante aiguille, avec de multiples fontes (appelables aussi par le traitement de texte A6), et un éditeur d'en-tête pour personnaliser un logo. Contient un spooler pouvant aller jusqu'à 400 K !

A9 : LE MONDE : La carte du monde s'affiche sur votre écran, avec zoom, recherche d'un endroit n'importe où sur le globe, passage des ouragans, etc... C.G.A.

A10 : JEUX : Empire, vous définissez ce que produiront vos cités (avions, troupes, bateaux, etc...), puis vous essayez de défendre ou d'attaquer des places ennemies, très complexes ; Blackjack, etc... C.G.A.

A11 : JEUX : Échecs, Pacman en 3D, Kong, etc... C.G.A.

A12 : JEUX : Othello, Flippeur, Dragons et Donjons, etc... C.G.A.

Lotus et D Base 3 sont des marques déposées ®

Bon de Commande à retourner à :

COMMUNICATIONS ET TECHNOLOGIES INFORMATIQUES
Centre Bonlieu - 74000 ANNECY

Je désire recevoir très rapidement les disquettes suivantes :

- *Ma première sélection de 4 disquettes :*

___ ___ ___ ___ **= 200 F + 16 F frais de port**

- *Ma seconde sélection de 4 disquettes :*

___ ___ ___ ___ **= 400 F + 16 F frais de port + Cadeau disquette de DAO**

☐ 3 1/2 ☐ 5 1/4

Nom _____

Prénom _____

Adresse _____

Code Postal _____ Ville _____

RENDRE COMPATIBLE SON IMPRIMANTE

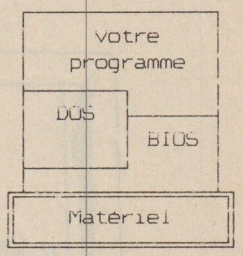
— INITIATION A L'ASSEMBLEUR —

Qui n'a jamais eu de problème de configuration d'imprimante, cette dernière ne comprenant pas les caractères IBM ? Sur ce thème, nous allons d'une part écrire un petit utilitaire pour transformer n'importe quel caractère à destination de l'imprimante par un autre, d'autre part en profiter pour voir toutes les bases de la programmation en assembleur.

Installer une imprimante compatible Epson ou IBM sur un compatible PC ne pose en général pas de problème, mais il y a le cas des imprimantes exotiques ou de celles qui diffèrent du standard de quelques caractères.
Le cœur du problème, c'est la codification des caractères. En micro-informatique, nous utilisons le code ASCII — ou code standard américain pour l'échange d'informations ; celui-ci définit 128 caractères pour les liaisons informatiques.
96 de ces caractères, codés de 32 à 127, représentent tous les symboles qu'un Américain peut avoir besoin d'afficher ou imprimer : les lettres, les chiffres, la ponctuation,...
Les 32 premiers caractères, de 0 à 31, les caractères de contrôle sont dits non affichables, ils servent à gérer le dialogue, entre deux installations informatiques. Par exemple, le code 13 correspond à la touche Enter ou encore à "retour à la ligne", le code 26 signifie "fin de fichier".
Sur les compatibles IBM, nous utilisons un code ASCII étendu allant de 0 à 255. La première partie reprend le code ASCII, et à partir de 128 nous trouvons d'autres symboles : lettres à accents, caractères semi-graphiques pour tracer des cadres, symboles mathématiques,...
Là où le bât blesse, c'est que les Américains ne se servent pas de lettres accentuées, et que des imprimantes ont été construites avant qu'IBM ne crée le Personal Computer. Nous trouvons donc sur le marché des imprimantes comportant un code ASCII altéré, où des symboles peu utilisés — de code inférieurs à 127, comme les crochets carrés [ ] — ont été remplacés par des lettres trançaises (é, ç, à,...). Ces imprimantes peuvent aussi faire jouer un rôle totalement différent aux codes supérieurs à 128, par exemple lettres en italiques ou mosaïque semi-graphique.
Ce que nous allons faire ici, c'est intercepter tous les caractères à destination de l'imprimante, et les remplacer par un caractère que l'imprimante comprendra.

Comment votre PC imprime-t-il ?

La traduction ne pourra être parfaite : en effet, nous ne pouvons remplacer 256 caractères par 128, mais nous pouvons faire une bonne approximation.
Vous dialoguez avec votre machine par l'intermédiaire d'un clavier, d'un écran, d'une imprimante, voire d'autres organes périphériques. Vous êtes conscient qu'à l'autre bout il y a un programme que vous utilisez : traitement de texte, tableur, etc. Et entre les deux, que se passe-t-il ?
Entre les deux, c'est la place du système, un ensemble de programmes de base destinés à gérer la machine. Sur un PC, le système est composé de deux parties, d'une part le **BIOS** ou système de base d'entrées-sorties, d'autre part le **DOS** ou système d'exploitation du disque. Le schéma est le suivant :



vous, l'utilisateur

Ce schéma illustre la façon dont le programmeur se sert du BIOS ou du DOS pour dialoguer avec vous au travers du matériel. Toutes les impressions passent par un même point situé dans le BIOS. Il suffit donc de se placer à cet endroit, de transformer les caractères et de les envoyer à l'ancien programme d'impression.
Pour ce faire, la meilleure technique consiste donc à programmer en assembleur, ce qui demande un minimum de connaissance de la machine.
Au cœur de l'ordinateur, se situe le **microprocesseur**. Pour un PC, il s'agit d'un 8088 et pour un AT, d'un 80286. Quelque part en mémoire, ils passent leur temps à lire des ordres et le programme, et à les exécuter. L'assembleur est le langage de

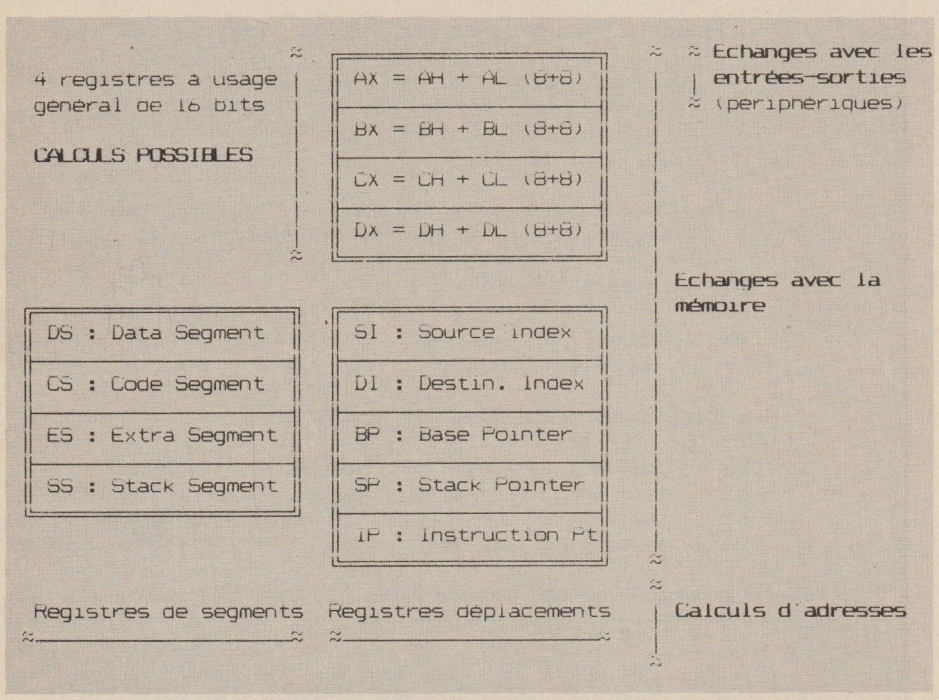
programmation le plus près possible de la machine : à une instruction assembleur correspond un ordre exécutable par le microprocesseur.

Organisation du 8088

Le 80286 étant un grand frère du 8088, il comprend tous les ordres exécutables par un 8088. Voyons maintenant comment est organisé un 8088. Toutes les informations manipulées par un 8088 passent par l'intermédiaire de mémoires internes à celui-ci : les registres.

Les registres sont de 16 bits : ils peuvent stocker des nombres entre 0 et 2^{16}-1, soit 65 535.

De prime abord, nous pourrions considérer le schéma fonctionnel d'un 8088 comme étant celui-ci :



Ce schéma montre plusieurs types de registres au cœur d'un 8088.

— Les quatre premiers registres, dits d'usage général, sont consacrés, entre autres, au calcul. Leur contenu peut être vu comme la juxtaposition de deux demi-registres de 8 bits, pouvant donc contenir une valeur entre 0 et 255. Le demi-registre de gauche est dit de poids fort (High en anglais), celui de droite de poids faible (Low), d'où leur nom AH et AL, BH et BL,... pour les demi-registres de AX, BX,...

— Les huit registres 16 bits suivant servent à calculer les adresses. En effet, le calcul d'une adresse nécessite plus de 16 bits. Pour cela, nous utilisons une première valeur 16 bits, appelée segment, nous la multiplions par 16, et y ajoutons une deuxième valeur 16 bits appelée déplacement, et nous notons cela segment : déplacement.

En examinant le nom de ces registres, vous devez vous douter qu'ils sont tous dédiés à des rôles différents.

CS:IP permet au microprocesseur de mémoriser où se trouve la prochaine instruction à exécuter (pointeur d'instruction dans le segment de code, c'est-à-dire de programme).

DS:?? en général l'adressage des données se fait avec DS contenant le segment de données.

DS:SI et **ES:DI** permettent de manipuler deux zones de mémoire en parallèle avec certaines instructions. L'une de ces zones est considérée comme source, l'autre comme destination (par exemple pour la comparaison de chaînes de caractères, ou de zone mémoire).

SS:SP pointe dans une zone mémoire un peu particulière : la pile (*stack* en anglais).

Empiler une donnée consiste à la mettre en mémoire au bout de cette zone, et à déplacer la limite de cette zone (en diminuant SP).

Dépiler une donnée consiste à lire la donnée à la limite de la pile et à déplacer cette limite en sens inverse. Certaines instructions assembleur font cela automatiquement : PUSH pour l'empilage, et POP pour le dépilage.

Dialogue avec le système

Jetons à nouveau un coup d'œil à notre premier schéma, cela pour attirer l'attention sur le fait que lorsque nous voulons dialoguer avec le matériel, nous devons passer par le système ; cela présente, entre autres, l'intérêt de nous éviter d'avoir à réécrire les fonctions de gestion du matériel, certainement parmi les plus complexes.

La question est «comment dialogue-t-on avec le système ?». La réponse est «par un jeu d'interruptions codifiées». Expliquons un peu cela. Dans un ordinateur comme le PC, il existe deux types d'interruptions.

— D'une part, nous avons les **interruptions matérielles**, résultat d'un organe électronique d'entrées-sorties qui envoie un signal au microprocesseur, «occupe-toi de moi» en quelque sorte. Ce signal est destiné à signaler une tâche urgente à traiter, comme la frappe d'un caractère sur le clavier. Le microprocesseur interrompt alors le programme en cours, sauvegarde son état dans la pile, et va exécuter un petit programme spécial pour régler le sort de ce caractère qui vient d'arriver. Une fois ce programme terminé, le microprocesseur retrouve son état antérieur à l'interruption grâce aux informations qu'il avait mises dans la pile, et reprend son travail original, là où il l'avait laissé.

Comme dans un PC plusieurs événements différents peuvent être la cause d'une interruption, ceux-ci sont identifiés par un numéro.

— Le deuxième type d'interruption, ce sont les **interruptions logicielles**, c'est-à-dire engendrées par programme grâce à l'instruction INT n°, où le numéro permet de faire la différence entre plusieurs interruptions. Les concepteurs du 8088 se sont rendu compte que ce pouvait être un moyen pratique de faire exécuter des bouts de programme sans qu'on ait besoin de connaître leur adresse.

Tous les appels système (BIOS ou DOS) se font par des interruptions logicielles codifiées : c'est en fait ce qui devrait définir la compatibilité DOS. Par exemple, la majeure partie des appels au DOS se fait au travers de l'interruption 33, soit 21h en hexadécimal. Auparavant, vous mettez un code dans le registre AH pour dire au DOS ce que vous voulez qu'il fasse, vous mettez d'éventuelles valeurs dans les autres registres, vous faites une instruction INT 21h, et vous avez passé la main au DOS, au retour vous pouvez trouver des réponses du DOS dans certains registres.

1er exemple : la fonction AH = 9 de l'interruption 21h permet d'afficher une chaîne de caractères se terminant par un '$' ; l'adresse de cette chaîne doit se trouver dans les registres DS:DX. Vous en avez deux illustrations dans le programme ci-joint, après les commentaires Termine le programme.

2e exemple : l'interruption 20h, avec CS ayant la même valeur qu'au démarrage du programme permet de terminer l'exé-

cution d'un programme et de rendre la main à l'interpréteur de commande du DOS : COMMAND.COM. Voir l'illustration à la fin de notre exemple.

Les fonctions BIOS d'impression

Tout ce qui concerne l'imprimante doit passer par l'interruption 23, ou 17h en hexadécimal, qui s'utilise de la manière suivante par un programme voulant commander une imprimante :

— Pour toutes les fonctions, mettre dans DX le numéro de l'imprimante concernée : 0 pour PRN ou LPT1, 1 pour LPT2, 2 pour LPT3.

— Mettre dans AH un code permettant de savoir ce que nous voulons faire : 0 pour imprimer le caractère dans AL, 1 pour initialiser la liaison avec l'imprimante, 2 pour connaître l'état de l'imprimante.

— Provoquer une interruption 17h grâce à l'instruction INT 17h.

— Après l'exécution de cette tâche, le BIOS donne un compte-rendu de l'opération dans le registre AH, si le bit 7 de AH vaut 1, il y a eu erreur.

Une part de ces informations est importante pour nous, car nous allons écrire un petit programme qui va rester résident, et que nous allons installer pour traiter l'interruption 17h.

Cette partie du programme se trouve dans notre listing, après le commentaire Pseudo-programme de service du BIOS. Dans ce programme, nous n'intervenons que sur les caractères à destination de PRN, c'est-à-dire pour DX = 0, et AH = 0. Pour les personnes qui ne sont pas familiarisées avec l'assembleur, nous avons joint un glossaire des instructions utilisées.

Si, par contre, il s'agit d'une écriture sur PRN, nous mettons dans DS:BX l'adresse d'une table permettant de remplacer le caractère contenu dans AL par le caractère correspondant dans la table, grâce à l'instruction XLAT. Si AL vaut 245, et si le 245e caractère de la table est 89, AL sera remplacé par 89. Nous reviendrons sur l'usage de cette table lors des directives d'utilisation de notre programme.

Une fois la substitution effectuée, nous faisons un saut jusqu'à l'ancienne adresse de traitement de l'interruption 17h, le module BIOS originel. Ainsi, après la traduction de caractère, la main est donnée au BIOS.

Installation du programme

La première instruction exécutable de notre programme est **JMP Installe**. Cette instruction demande de continuer l'exé-

```
PAGE ,132
;-----------------------------------------------------------------;
; PrnXlat.Asm : Traduit les caracteres transmis a l imprimante   ;
;                                                                 ;
;                        Jean-Paul Pruniaux, le 18 septembre 1988;
;-----------------------------------------------------------------;

Code    SEGMENT  CODE
        ASSUME   CS:Code, DS:Code, ES:Code

        ORG      100h

Debut:  JMP      Installe

; Zone de donnees
; ---------------

        ; Pour memoriser l adresse de l ancienne Int 17h
Old_17h DD       0

Table:
        ; Premiere partie de la table ASCII : non visualisable

        DB   00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
        DB   08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
        DB   10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
        DB   18h, 19h, 1Ah, 1Bh, 1Ch, 1Dh, 1Eh, 1Fh

        ; Deuxieme partie de la table ASCII : visualisable
        ; (remarque 27h est le code de l apostrophe)

        DB    !"#$%& ,27h
        DB              ()*+,-./ ,  01234567 ,  89:;<=>?
        DB   @ABCDEFG ,  HIJKLMNO ,  PQRSTUVW ,  XYZ[\]^_
        DB   abcdefg ,  hijklmno ,  pqrstuvw ,  xyz{|}~ , 7Fh

        ; Caracteres graphiques IBM (jeu etendu)

        DB   ÇüéâäàåÇ ,  êëèïîìÄÅ ,  ÉæÆôöòûù ,  ÿÖÜ¢£¥₧ƒ
        DB   áíóúñÑªº ,  ¿⌐¬½¼¡«» ,  ░▒▓│┤╡╢╖ ,  ╕╣║╗╝╜╛┐
        DB   └┴┬├─┼╞╟ ,  ╚╔╩╦╠═╬╧ ,  ╨╤╥╙╘╒╓╫ ,  ╪┘┌
        DB   αßΓπΣσµτ ,  ΦΘΩδ∞∅∈∩ ,  ≡±≥≤⌠⌡÷≈ ,  °∙·√ⁿ²■ , 0FFh

; Pseudo programme de service du BIOS
;-----------------------------------
Int_17:
        OR       DX, DX          ; Port 0 ?
        JNE      Fin_17          ; non : donner la main au BIOS
        OR       AH, AH          ; Ecriture ?
        JNE      Fin_17          ; non : donner la main au BIOS

        PUSH     DS              ; Sauve DS
        PUSH     CS
        POP      DS              ; DS = CS
        PUSH     BX              ; Sauve BX

        MOV      BX, OFFSET Table; DS:BX = Adresse de la table
        XLAT                     ; Remplace le code dans AL par
                                 ; celui correspondant de la table

        POP      BX              ; Restitue BX
        POP      DS              ; Restitue DS

Fin_17:                          ; Fin en donnant la main au BIOS
        JMP      DWORD PTR CS:[Old_17h]

Fin_Res:                     ; derniere adresse residente
Lng_Res     EQU $ - Int_17   ; longueur residente
```

```
        ; Messages d'installation
        ; ------------------------

        Msg_Pre DB  'PrnXLat.Com déjà en mémoire. ', 0Dh, 0Ah, '$'
        Msg_Ins DB  'Installation de PrnXLat.Com. ', 0Dh, 0Ah, '$'

        ; Initialisation
        ; --------------
        Installe:

        ; Lecture de l'ancien vecteur INT 17h
        ; -----------------------------------

        MOV     AL, 17h                 ; Numéro d'interruption
        MOV     AH, 35h                 ; Code demande adresse traitement
        INT     21h                     ; Appel au DOS
        MOV     WORD PTR [Old_17h],     BX     ; sauve l'adresse
        MOV     WORD PTR [Old_17h+2], ES

        ; Vérifie si déjà installé
        ; ------------------------

        MOV     DI, Bx                  ; ES:DI = Adr ancienne routine
        MOV     SI, OFFSET Int_17       ; DS:SI = Adr de ce programme
        MOV     CX, Lng_Res             ; CX = Taille de ce résident
        CLD                             ; Initialise la comparaison
        REPE    CMPSB                   ; compare les deux programmes
        JE      Deja_La

        ; Termine le programme et reste résident
        ; --------------------------------------

        MOV     DX, OFFSET Msg_Ins      ; DS:DX = Adresse du 1er Message
        MOV     AH, 09h                 ; Code d'envoi de message
        INT     21h                     ; Appel DOS

        MOV     AL, 17h                 ; Numéro d'interruption
        MOV     DX, OFFSET Int_17       ; DS:DX = adresse de notre pgm
        MOV     AH, 25h                 ; Code installation de notre pgm
        INT     21h                     ; Appel DOS

        MOV     DX, OFFSET Fin_Res      ; DX = Adresse fin du résident
        MOV     CL, 4
        SHR     DX, CL                  ; division par 16
        INC     DX                      ; arrondi à seizaine supérieure
        MOV     AX, 3100h               ; code fin de pgm reste résident
        INT     21h                     ; donne la main au DOS

        ; Termine le programme sans rester résident
        ; -----------------------------------------
Deja_La:

        MOV     DX, OFFSET Msg_Pre      ; DS:DX = Adresse du 2e message
        MOV     AH, 09h                 ; Code d'envoi de message
        INT     21h                     ; Appel DOS

        Int     20h                     ; Fin de programme

Code    ENDS
        END     Debut
```

cution du programme à la ligne marquée Installe :.

— La première chose à faire, c'est de dialoguer avec le DOS au travers de l'interruption 21h pour savoir où est traitée l'interruption 17h. Nous avons besoin de cette adresse afin de terminer notre programme résident par un saut à cet endroit. Avant l'interruption 21h, il nous faut mettre dans AH le code 35h, et dans AL le numéro de l'interruption en question. Le DOS nous communique cette adresse au travers des registres ES et BX.

— Ensuite, il faut vérifier que notre programme n'a pas déjà été installé antérieurement. Pour cela, il suffit de comparer le contenu de la mémoire à l'adresse que vient de nous communiquer le DOS avec celui de notre programme en CS : mInt17h.

Pour comparer deux contenus de mémoire, nous avons deux instructions assembleur : REPE COMPSB, ou répétition de comparaison de chaînes, octet par octet. Pour utiliser cette instruction, il nous faut d'abord initialiser DS:SI et ES:DI afin qu'ils contiennent les adresses de ces deux zones de mémoire. D'autre part, il faut dire dans quel sens à partir de ces adresses, il faut faire la comparaison (CLD comparaison par adresse montante), et finalement, il faut mettre dans CX le nombre d'octets à comparer.

Si nous nous rendons compte que le programme est déjà installé, inutile, voire grave de le réinstaller, nous faisons un saut à la ligne marquée Dejala :, pour afficher un message en conséquence, et pour terminer le programme de manière ferme et définitive.

En revanche, si notre programme n'est pas encore installé, nous allons faire les deux choses suivantes :

— demander au DOS d'installer notre programme pour traiter l'interruption 17h,

— terminer le programme en demandant au DOS de nous en laisser une partie en mémoire.

Pour demander au DOS d'installer un programme afin de traiter une interruption, nous utilisons la fonction 25h après avoir mis dans AL le numéro de l'interruption, et dans DS:DX sa nouvelle adresse de traitement.

Pour terminer un programme et rester résident, nous avons recours à la fonction AX = 3100h du DOS. Pour cela, nous devons aussi mettre dans DX le nombre de paragraphes (par groupe de seize) qui doivent rester en mémoire, d'où notre petit calcul.

Réalisation du programme

Ce qui suit décrit les opérations destinées à ceux qui possèdent un Assembleur ; que les autres se réfèrent à la fin de l'article.

Bien que nous ne vous l'ayons pas dit, notre intention est de construire un programme de type .COM. Nous avons maintenant toutes les instructions pour l'écrire, mais il nous reste encore quelques détails à préciser à l'assembleur.

Au démarrage d'un programme de ce type, tous les registres de segment sont à la même valeur, celle de CS. La première instruction à exécuter se trouve à l'adresse 100h dans le segment CS (autrement dit IP vaudra 100h), et aucune instruction ne doit se trouver avant cette adresse.

Nous donnons à l'assembleur l'ordre de

compter les adresses à partir de 100h grâce à la directive ORG 100h. (Une directive est un ordre donné à l'assembleur, par exemple, PAGE, 132 lui demande de présenter le listing en 132 colonnes).

Nous sommes aussi censés décrire à l'assembleur le contenu des registres de segment, pour qu'il puisse faire un certain nombre de vérifications. C'est ce que nous faisons avec les deux directives SEGMENT et ASSUME.

SEGMENT permet de caractériser un segment que nous avons baptisé Code, c'est du programme ; **ASSUME**, lui, signale que nous utilisons les registres CS, DS et ES, et que le contenu de ces trois registres est *a priori* la même valeur. Comme nous avons utilisé une directive SEGMENT, nous devons aussi signaler la fin de ce segment grâce à la directive **ENDS**. Cette dernière correspondant à la

● La syntaxe d'une ligne assembleur est la suivante :
{Symbole} Instruction {Source} {,Destination} {; Commentaires}

● Types d'instructions utilisées dans PrnXLat.Asm.

: (exemple Debut:) — Symbole servant à associer un emplacement dans le programme (un déplacement) à un nom de notre invention. Ce nom pourra être utilisé par exemple lorsque nous voudrons faire un saut à cette adresse.

ASSUME — Directive d'assemblage permettant de signaler à l'assembleur quel est le contenu des registres de segment. Les segments doivent avoir été antérieurement définis par une directive SEGMENT.

CLD — Instruction demandant à ce que les opérations de manipulation de chaîne de caractères ou de blocs de mémoire se fassent avec des adresses croissantes (voir REPE COMPSB).

DB '@ABCDEFG' — Define Byte, définition d'octets. Cette directive demande à l'assembleur de réserver de la place en mémoire à cet endroit pour mettre les octets dont la liste suit. Ici, en l'occurrence, les octets sont les codes ASCII des symboles **@, puis A, puis ...**

DB 00h, 01h — idem, mais les octets sont définis par leur valeur hexadécimale 0, puis 1, ...

Msg_Pre DB 'Pr...', 0Dh, 0Ah, '$' — idem, avec utilisation des deux possibilités précédentes. De plus, ces octets sont précédés d'un nom de notre propre cru qui permettra d'y faire référence ultérieurement.

Old_17h DD 0 , Define Double word — Définition d'un double mot, soit 32 bits. Cette instruction demande à l'assembleur de réserver quatre octets en mémoire. Nous y associons un nom. Quatre octets sont nécessaires par exemple pour sauvegarder une adresse. Les deux premiers octets contiennent alors le déplacement, et les deux suivants le segment.

END Debut — Directive signalant à l'assembleur que c'est la fin du programme, et que la première instruction à exécuter est celle de la ligne Debut:.

Code ENDS — Directive signalant la fin du segment Code défini par une directive Code SEGMENT....

INC DX — Incrément de DX, autrement dit, ajoute 1 au contenu de DX.

INT 21h — Interruption logicielle numéro 21h. (correspond à un appel DOS).

INT 20h — Interruption logicielle numéro 20h. (Fin de programme). JE Deja_La : Jump on Equal, Saut si Egal. Si le résultat de la dernière opération est nul, ou si le résultat du dernier test est l'égalité, alors sauter pour aller exécuter les instructions marquées par Deja_La:.

JMP Installe — Saut pour aller exécuter les instructions marquées par Installe:

JMP DWORD PTR CS:[O ld_17h — Saut pour aller exécuter les instructions dont l'adresse se trouve en mémoire en CS:Old_17h. Autrement dit, dans le segment de code CS, se trouvent quatre octets qui contiennent l'adresse du programme que nous voulons exécuter maintenant.

Lng_Res EQU ... — Equate, Egale à, nous définissons un symbole et sa valeur est la suivante...

Lng_Res EQU $ - Int_17 — $ représente l'adresse à laquelle nous sommes en train d'assembler, le résultat de l'opération $-Int_17 est donc le nombre d'octets compris entre l'emplacement de Int_17: et l'emplacement actuel.

MOV — Move, déplacement d'informations, le premier élément qui suit le MOV dit où va la donnée, le deuxième, d'où elle vient.

MOV WORD PTR [O ld_17h], BX — Met en mémoire à l'adresse DS:Old_17h le mot contenu par le registre BX.

MOV AH, 35h — Met dans le registre AH la valeur hexadécimale 35h.

MOV DX, OFFSET Msg_Pre — Met dans le registre DX le déplacement (offset) correspondant à Msg_ Pre. Ce dernier est un symbole que nous avons défini avec un DB.

MOV CX, Lng_Res — Met dans DX la valeur Lng_Res, celle-ci a été calculée par un EQU.

MOV DI, BX — Met dans le registre DI, le contenu du registre BX.

OR CX, DX — Fait un OU logique entre CX et DX. Pour faire un OU entre deux valeurs, on regarde ces valeurs bit à bit, et si l'un des deux bits est à 1, on reporte 1 dans le résultat. Exemple :

1100 1010
ou 0101 0110
= 1101 1110

OR DX,DX ne va pas changer le contenu de DX mais permet de tester si celui-ci est nul. Nous aurions pu écrire CMP DX,0, autrement dit, compare le contenu de DX avec 0, il s'agit là d'une vieille habitude de programmeur (il y a aussi une autre raison : le programme sera moins long d'un octet).

ORG 100h — Directive d'assemblage signifiant que le déplacement qui suit vaut 100h.

PAGE ,132 — Directive d'assemblage pour présenter le listing sur 132 colonnes.

POP DS — Dépile le contenu de DS.

PUSH DS — Empile le contenu de DS.

REPE CMPSB — Compare deux zones de mémoire pointées par DS:SI et ES:DI, voyez l'article.

Code SEGMENT 'CODE' — Définition du segment de code : nous lui donnons le nom Code, et il correspond au segment de code, c'est-à-dire au programme.

SHR DX, CL — Décalage de CL bits du registre DX sur la droite. Exemple CL = 2 :
avant le décalage DX = 1110 0101 0110 0111
après le décalage DX = 0011 1001 0101 1001
Notez qu'un décalage de 1 bit à droite est l'équivalent d'une division par 2 (c'est de l'algèbre binaire).

XLAT — Traduction (Translate), DS:BX contient l'adresse d'une table, et AL le caractère à traduire, voyez l'article.

DEBUG : CRÉER LE PROGRAMME SANS ASSEMBLEUR

```
C>DEBUG
-A 100
291E:0100 JMP 025C
291E:0103 DW 0,0
291E:0107 DB 00,01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F
291E:0117 DB 10,11,12,13,14,15,16,17,18,19,1A,1B,1C,1D,1E,1F
291E:0127 DB ' !"#$%&',27
291E:012F DB              '()*+,-./', '01234567', '89:;<=>?'
291E:0147 DB '@ABCDEFG', 'HIJKLMNO', 'PQRSTUVW', 'XYZ[\]^_'
291E:0167 DB '`abcdefg', 'hijklmno', 'pqrstuvw', 'xyz{¦}~' , 7F
291E:0187 DB 80,81,82,83,84,85,86,87,88,89,8A,8B,8C,8D,8E,8F
291E:0197 DB 90,91,92,93,94,95,96,97,98,99,9A,9B,9C,9D,9E,9F
291E:01A7 DB 'áíóúñÑªº', '¿⌐¬½¼¡«»', '',
291E:01C7 DB '',
291E:01E7 DB 'αβΓπΣσμτ', 'ΦΘΩδ∞φ∈∩', '≡±≥≤⌠⌡÷≈', '°∙·√ⁿ²■',  FF
291E:0207
-A 207
291E:0207 OR DX,DX
291E:0209 JNZ 0219
291E:020B OR AH,AH
291E:020D JNZ 0219
291E:020F PUSH DS
291E:0210 PUSH CS
291E:0211 POP DS
291E:0212 PUSH BX
291E:0213 MOV BX,0107
291E:0216 XLAT
291E:0217 POP BX
291E:0218 POP DS
291E:0219 CS:
291E:021A JMP FAR [0103]
291E:021E DB  'PrnXLat.Com déjà en mémoire.', 0D, 0A, '$'
291E:023D DB  'Installation de PrnXLat.Com.', 0D, 0A, '$'
291E:025C MOV AL,17
291E:025E MOV AH,35
291E:0260 INT 21
291E:0262 MOV [0103],BX
291E:0266 MOV [0105],ES
291E:026A MOV DI,BX
291E:026C MOV SI,0207
291E:026F MOV CX,0017
291E:0272 CLD
291E:0273 REPZ
291E:0274 CMPSB
291E:0275 JZ 0294
291E:0277 MOV DX,023D
291E:027A MOV AH,09
291E:027C INT 21
291E:027E MOV AL,17
291E:0280 MOV DX,0207
291E:0283 MOV AH,25
291E:0285 INT 21
291E:0287 MOV DX,021E
291E:028A MOV CL,04
291E:028C SHR DX,CL
291E:028E INC DX
291E:028F MOV AX,3100
291E:0292 INT 21
291E:0294 MOV DX,021E
291E:0297 MOV AH,09
291E:0299 INT 21
291E:029B INT 20
291E:029D
-N PrnXLat.Com
-RCX
CX 0000
:19D
-
-W
019D octets écrits
-Q
C>PrnXLat
```

fin du programme, nous lui faisons suivre une autre directive END qui non seulement signale la fin du programme à l'assembleur, mais en plus nous permet de lui dire où se trouve la première instruction de ce programme. Comme nous faisons un programme de type .COM, nous devons obligatoirement faire référence à la première instruction qui suit la directive ORG 100h.

Avec ce que nous venons de voir, la structure d'un programme .COM est toujours similaire à celle du programme trivial suivant :

```
C:\ASM>TYPE TRIVIAL.ASM
Code    SEGMENT PARA PUBLIC 'CODE'
        ASSUME  CS:Code, DS:Code, ES:Code
        ORG     100h

Debut:                  ; Début du programme
        INT     20h
Code    ENDS
        END     Debut
C:\ASM>
```

Maintenant que nous avons écrit notre programme, il ne nous reste plus qu'à le faire réaliser par la machine. Cela se passe en plusieurs étapes :

— d'abord l'assemblage qui nous fournit un programme TRIVIAL.OBJ ;

— la deuxième étape est l'édition de lien, elle nous permet de regrouper plusieurs programmes OBJ ou de leur ajouter des fonctions écrites dans des librairies, le résultat de ce travail est un module EXE. Si nous avions écrit notre programme pour qu'il puisse être de type EXE nous

Après le listing de notre programme PrnXlat.Asm, nous avons donné un listing permettant de créer le programme grâce à DEBUG.

En effet, DEBUG, livré avec le DOS possède une commande pour assembler directement en mémoire. Pour cela, derrière l'indicatif du DOS, A>, B> ou C>, tapez DEBUG. Celui-ci répond par un petit tiret. Tapez alors A pour assembler, DEBUG répond par une adresse ????:0100, tapez alors vos instructions d'assemblage conformément à notre listing.

A la fin de ce listing, la commande N sert à signaler à DEBUG le nom du fichier que vous voulez créer. Nous modifions alors le contenu du registre CX pour indiquer la taille du fichier, avant de lancer la commande W pour écrire ce que nous venons d'assembler sur le disque. Comme les 256 premiers octets, 100h, ne sont pas écrits sur le disque, la longueur à mettre dans CX est la dernière adresse d'assemblage diminuée de 100h.

Vous pouvez modifier notre table de traduction conformément à ce que nous avons vu avec l'utilisation de l'assembleur, mais vous pourrez aussi le modifier après coup de la manière suivante : chargez le programme sous DEBUG grâce à la commande suivante DEBUG PrnXLat.Com, puis allez modifier la table en mémoire avec la commande E.

Cette table commence à l'adresse 0107. Supposons que vous vouliez modifier le code 7Bh. Il vous faudra éditer le code à l'adresse 107h+7Bh soit 182h, vous tapez alors E 182, puis vous donnez la nouvelle valeur hexadécimale à mettre dans la table. Pour terminer, vous faites un W pour mettre à jour le fichier.

MICRO 2

16, Rue des Fossés
35000 RENNES
TEL 99.63.71.11

LOGICIELS EDUCATIFS

TYPE DE MACHINE	PC 5"1/4	PC 3"1/2	CPC	ST
EDUC-MATERNELLE-1	220	240	200	220
EDUC-MATERNELLE-2			200	
LECTURE-CP			200	
MATHS-CE	220	240	200	220
MATHS-CM	240	240	240	240
CALCUL-RAPIDE			200	
EDUC-PRIMAIRE				220
FRANCAIS-SONS	220	240	200	220
ORTHO-CM	220	240	200	220
FRANCAIS-CM	220	240	200	220
GEOGRAPHIE-PRIMAIRE			200	
GUILLOTINE				220
GRAMMAIRE-65	220	240	200	220
MATHS-6	220	240	200	220
MATHS-5	220	240	200	220
MATHS-54			200	220
MATHS-4	220	240		
MATHS-3	220	240	200	220
MATHS-2	240	240		240
MATHS-1	240	240		240
MATHS 2ND CYCLE-1			200	
MATHS 2ND CYCLE-2			200	
GEOMETRIE			200	220
HISTOIRE-PRIMAIRE				220
ANGLAIS-43				240
REVOLUTION-89	260	260		240

Voltaire

Robespierre

Danton

Rousseau

BON DE COMMANDE

TITRE	TYPE	QTE	PRIX

REMISE 10% POUR 3 LOGICIELS ET PLUS

NOM

ADRESSE

CODE POSTAL VILLE

INITIATION

aurions pu nous arrêter là, nous aurions un module exécutable. Nous avons préféré faire un module COM pour plusieurs bonnes raisons, l'une d'entre elles étant qu'ils sont plus simples à écrire. Nous devons donc transformer ce programme EXE en COM grâce à l'utilitaire du DOS EXE2BIN.

Au passage, lors de l'édition de lien, LINK va et doit nous sortir un message : Warning, no stack segment. C'est normal, cela vient du fait que nous faisons un module COM.

La procédure est la suivante :

```
C:\ASM >>> masm Trivial;
Microsoft (R) Macro Assembler  Version 4.00
Copyright (C) Microsoft Corp 1981, 1983, 1984, 1985.  All rights reserved.


  49864 Bytes symbol space free

       0 Warning Errors
       0 Severe  Errors

C:\ASM >>> link Trivial;

Microsoft (R) 8086 Object Linker  Version 3.05
Copyright (C) Microsoft Corp 1983, 1984, 1985.  All rights reserved.

Warning: no stack segment
C:\ASM >>> exe2bin Trivial.exe Trivial.com
C:\ASM >>> del Trivial.exe
C:\ASM >>> Trivial
C:\ASM >>>
```

Utilisation du programme

Le programme PrnXlat.Asm dont vous avez le listing n'est en fait pas du tout paramétré pour votre imprimante. En effet, nous ne connaissons pas *a priori* votre imprimante.

La première chose à faire consiste donc à corriger la table en conséquence, et pour cela vous devez savoir quel code imprime quel symbole. Cela doit normalement se trouver dans le manuel de votre imprimante. Si toutefois vous avez des doutes, ou si vous n'êtes pas sûr de la configuration de votre imprimante, vous pouvez toujours avoir recours au petit programme Basic ci-dessous qui imprimera tous les caractères possibles avec leur code :

```
FOR I = 32 TO 255 : LPRINT I, HEX$(I),
        CHR$(I) : NEXT I
```

Maintenant, il vous faut analyser la liste obtenue en comparaison avec les codes ASCII étendus par IBM. Par exemple, vous allez vous rendre compte que le é accent aigu, codé 130 ou 82h par IBM, est en fait obtenu sur votre imprimante en lui envoyant un code 123 ou 7Bh. Cela veut dire deux choses : d'une part lorsque vous voulez imprimer un é il faut envoyer 123 ou 7Bh à l'imprimante, d'autre part, l'accolade ne doit probablement pas être définie, il lui faudra certainement un caractère de substitution (par exemple une parenthèse).

Pour remplacer le code du é dans notre table, vous allez procéder de la manière suivante :

```
; Caractères graphiques IBM (jeu étendu)

DB    Cu , 7Bh
DB        aaaaç ,  eeèïïAA ,  Eæ¶oooûù ,  yOU¢£ƒ¶ƒ
```

Nous avons également changé les a accents graves qui le suivaient en a.

Une fois le source du code modifié en conséquence, vous préparez le programme comme nous l'avons vu au paragraphe précédent (en remplaçant le mot Trivial par PrnXLat), puis vous tapez la commande PrnXLat. Vous devez obtenir le message d'installation, puis toutes les impressions qui suivent doivent être corrigées comme vous l'avez fait.

La balle est maintenant dans votre camp...

Jean-Paul Pruniaux

LES FICHIERS EN ACCES DIRECT

Le Basic peut exploiter un fichier énorme, des centaines de kilo octets, et ce par le biais de "l'accès direct". Cela sous-entend qu'il reste sur disque et que l'on ne charge en RAM que l'enregistrement (la "fiche") dont on a besoin ; il suffit d'en connaître le numéro.

Idem pour la mise à jour d'une fiche ; seule celle-ci est re-écrite, et à l'emplacement même de l'ancienne. Le terme "accès direct" est donc bien justifié. Il n'est plus nécessaire de re-enregistrer tout le fichier comme c'est le cas avec les "fichiers séquentiels". En prime et à titre d'exemple concret, nous vous livrons un programme de gestion de fichiers qu'il vous sera très facile d'adapter à vos besoins

Comparaison avec les fichiers séquentiels.

Généralement un fichier séquentiel sert à garnir un ou plusieurs tableaux DIM ; on est donc limité par la RAM disponible qui en Basic ne dépasse pas 60 kilo octets. En pratique le recours à l'accès direct est recommandé quand le fichier dépasse 20 ko environ (plus si le programme est compilé). - Rechercher une valeur dans un tableau DIM est ultra rapide, la recherche d'une valeur dans un fichier sur disque l'est beaucoup moins, à cause des "temps d'accès" dûs à la mécanique des bras de lecture. Pour cette raison le disque dur est ici préférable au lecteur de disquettes.. - Le risque de "garbage" (saturation de la RAM) par le renouvellement des variables chaînes devient négligeable en accès direct. - Tout est calibré à l'avance : on a fixé une longueur pour chaque "champ" (rubrique de la fiche). Si la valeur est courte elle sera complétée par des espaces, si elle est trop longue le Basic la tronquera sans préavis. C'est du rigoureux ! - Le fichier sur disque est monobloc ; un seul enregistrement mais pouvant faire des centaines de kilo octets. Tout y est écrit à la queue leu leu, sans le moindre retour ligne ou signe de séparation. Quand sous DOS on fait un TYPE sur un de ces fichiers celà peut paraître "pagaille"! Erreur ! c'est en fait super ordonné : On a indiqué la longueur LEN d'une fiche (le pas en octets), donc si on demande de lire ou écrire la fiche numéro 832 le bras saura aller directement dessus. - Une telle technique peut déboucher sur des applications n'ayant aucun rapport avec la gestion de fichiers, comme l'exploitation par Basic de fichiers même binaires issus de certains logiciels, et ce afin de les "bidouiller" à notre guise. On en parlera en annexe...

Ouverture d'un fichier en accès direct.

Prenons un exemple très simple. Trois "champs", nom, prénom et date de naissance. Il faut tout d'abord fixer la longueur de chaque champ. Cette instruction FIELD est obligatoire. Disons nom=20, prénom=15 et date=6 (en format AAMMJJ, 590304 pour le 4 mars 1959). Donc une fiche aura pour longueur LEN=20+15+6 = 41 Cette "réservation" est fixe et définitive. Si prénom = Paul il sera donc suivi par 11 espaces (ou "blancs") pour le compléter à 15. Gaspillage d'octets ? Certes, mais uniquement sur la disquette de 360 ko ou sur le disque dur.

```
OPEN "CLUB" AS #1 LEN=41
```

Deux remarques : Plus de "FOR INPUT" ou "FOR OUTPUT" ou "FOR APPEND", c'est ouvert à la fois en entrée et en sortie. On peut spécifier "FOR RANDOM" mais ce n'est pas obligatoire. En informatique le mot fiche est incorrect, il faut dire "enregistrement". Je conserverai "fiche" pour être plus clair.
A présent définissons les "segments champs" : (= longueurs et noms des "rubriques")

```
FIELD #1,20 AS NOM$, 15 AS PREN$,
6 AS DAT$
```

Un sage conseil : Ces 3 variables chaînes définies dans notre FIELD ont un statut assez privilégié, car elles ont désormais une zone réservée dans la mémoire RAM. N'utilisez plus ces noms ailleurs pour d'autres choses. Aucun risque de "garbage" ; ce sont des "variables FIELD", la haute noblesse...

Lecture et écriture d'une fiche.

Pour lire c'est GET (= prendre), pour écrire c'est PUT (= mettre, poser). Lisons la fiche N° 63

```
GET #1,63  :' prend dans le
fichier ouvert en #1 la fiche 63
```

Que fait le PC ? Au départ du fichier "CLUB" il "saute" 62 fois 41 octets, et lit seulement les 41 suivants en les recopiant dans les cases mémoires définies par FIELD : les 20 premiers dans NOM$, les 15 suivants dans PREN$ et les 6 derniers dans DAT$. A présent nous pouvons afficher

```
PRINT NOM$ --> ANGREVE : PRINT
PREN$ --> Mélusine : PRINT DAT$ --
> 670401
```

Modifions DAT$ et sauvegardons cette mise à jour de fiche.

```
INPUT "Date de naissance AAMMJJ
",D$ LSET DAT$=D$ PUT #1,63
```

Expliquons : - Pour l'INPUT on prend un autre nom de variable, D$ - LSET vient de LEFT SET = "le contenu de DAT$ en partant de la gauche est D$" - PUT re-écrit toute la fiche N°63. Vous remarquez qu'il n'y aura plus de sous-programme pour la sauvegarde.

Il existe aussi RSET (RIGHT SET = à partir de la droite) mais d'un usage moins courant.

INPUT "Date ",DAT$ serait une faute grave mais sans message d'erreur : la "noble variable FIELD" DAT$ ne serait alors ni concernée ni modifiée ! Seuls GET, LSET et RSET peuvent modifier le contenu d'une variable FIELD. Il faudra bien s'en souvenir...

RESUMONS : Les fichiers en accès direct disposent de six commandes

Basic, à savoir LEN, FIELD, GET, PUT, LSET et RSET. (prononcez lène, filde, guette, poute, elsette et ersette). Plus certaines concernant les variables FIELD numériques d'un usage assez délicat ; pour débuter entrez les nombres sous forme de chaînes. Ne pas confondre GET et PUT avec leurs homonymes des commandes graphiques ni la commande LEN avec la fonction LEN(A$)... (rappel, une commande est théoriquement destinée au système d'exploitation, une instruction ordonne, modifie quelque chose, une fonction retourne une valeur et une déclaration ne fait que renseigner).

Quelques remarques.

A - La numérotation des fiches ne commence pas à zéro mais à UN ; PUT (ou GET) #1,0 = plantage.

B - La taille en octets d'un fichier en accès direct est exactement un multiple de LEN, ce qui permet d'en connaître le nombre de "fiches". NF=LOF(1)/41: ' nombre de fiches PRINT "Le fichier du CLUB a";NF;"adhérents"

```
LOF(1) = "LENGTH OF FILE", lon-
gueur du fichier ouvert en #1
```

C - Ne vous étonnez pas si chaque PUT ne provoque pas toujours une mise en route du drive. En effet le micro attend qu'il y ait au moins 512 octets à écrire (= 1 secteur de disque) ; toutefois il écrira ces 41 octets si c'est la fin, due à un CLOSE #1 ou END ou RUN.

D - On ne peut pas supprimer une fiche, on peut seulement la mettre "à blanc", quitte à ré-utiliser ce numéro de fiche plus tard.

E - Pour changer l'ordre des fiches il faudrait créer un autre fichier, "CLUB2" ouvert en #2, le remplir dans l'ordre souhaité, effacer "CLUB" (par KILL "CLUB") et enfin renommer "CLUB2" en "CLUB" par NAME "CLUB2" AS "CLUB".

La grande bêtise (hélas classique)

Nous avions fait un GET sur la fiche N° 63, donc les trois variables FIELD sont "garnies de valeurs fraîches". On a modifié DAT$ et on a sauvegardé par PUT la fiche de Mlle ANGREVE ; ça c'est OK.

A présent on veut faire la même modification mais sur la fiche de Mr BAVE Jean qui a le numéro 185. "Pas la peine de faire un GET, je suis sûr du numéro !", se dit le gaffeur, alors INPUT D$; LSET DAT$=D$ et PUT #1,185. Et voila la grosse bourde est faite : En effet en l'absence d'un GET ou de LSET les autres variables NOM$ et PREN$ contiennent toujours ANGREVE et Mélusine... et ce PUT va copier ces valeurs dans l'enregistrement N° 185 ! Donc ANGREVE sera en double en numéros 63 et 185 (avec des DAT$ différents) et le pauvre Mr BAVE aura été effacé du fichier !

D'où cette grande règle de sécurité :

En modifications de fiches, il faut toujours programmer un GET avant de faire un PUT.

Souvenez-vous que PUT prépare son tronçon de longueur LEN en mettant bout-à-bout les contenus actuels des "variables FIELD", puis il re-écrit toute la fiche sur le disque à l'emplacement exact de l'ancienne version. Donc cette dernière n'est plus récupérable en cas d'erreur. En accès direct il est prudent d'avoir un duplicata du fichier. En fin de programme programmez SHELL "COPY CLUB CLUB.BAK". On ne sait jamais...

La pratique du field.

La commande FIELD fait démarrer le "tronçonnage" au début de la longueur définie par LEN. Cela vous paraît évident, mais cela implique en fait que toutes les déclarations de champs doivent être sur une ligne unique commençant par "FIELD" ! Or une ligne Basic ne peut excéder 255 caractères... Et si l'on a une quarantaine de champs ?... Il faut ruser :

- Quand la première ligne FIELD est "pleine" on en crée une deuxième qui va commencer par un "champ talon", bidon, dont la longueur sera égale à la somme des champs déjà définis. Un exemple

Soit sept champs C1$ à C7$, chacun de longueur dix caractères. Pour être plus clair nos lignes sont très courtes.

```
810 OPEN "TRUC" AS #2 LEN=70 820
FIELD #2,10 AS C1$,10 AS C2$,10 AS
C3$ 830 FIELD #2,30 AS X1$,10 AS
C4$,10 AS C5$ 840 FIELD #2,50 AS
X2$,10 AS C6$,10 AS C7$
```

Ligne 820 on totalise 30 octets (C1$ à C3$). Ligne 830 on redémarre à zéro mais avec un talon X1$ de 30, on poursuit alors le tronçonnage avec C4$ et C5$; ce qui nous amène à 50 octets. Ligne 840 nouveau départ à zéro, mais avec un talon X2$ de 50, etc...

Pour la suite de l'exploitation du fichier on ignorera l'existence de ces champs talons : Nos LSET ne concernerons que les variables C1$ à C7$, et tout se passera sans le moindre ennui. Garanti. On peut même ajouter ceci :

```
850 FIELD #2,70 AS TOTAL$
```

Ce "champ global" sera utile pour vider d'un seul coup le contenu de toute une fiche, par exemple la N° 127 :

```
LSET TOTAL$="":PUT #2,127
```

Remarquez que nos lignes de FIELD suivent une progression logique de "gauche à droite" dans la fiche, et que TOTAL$ a de ce fait été déclaré en dernier. Faites de même et vous n'aurez jamais de chocs en retour.

Cas des champs nombreux.

Lorsque les champs sont nombreux on choisit en fait des noms de variables FIELD indicées ; mais n'oubliez pas de les déclarer en DIM. Un exemple avec 48 champs : 100 DIM R$(48),LR(48) LR = longueurs respectives (entrées en lignes de DATA)

```
250 LT=0:RESTORE 210:FOR I=1 TO
48:READ LR(I):LT=LT+LR(I):NEXT
```

La programmation des FIELD sera simple et courte en utilisant une boucle FOR NEXT et des "champs talons" X$.

```
3010 OPEN "MACHIN" AS #1 LEN=LT
3020 FIELD #1,LR(1) AS
R$(1):TT=LR(1):X=1 3030 FOR J=2 TO
40 3040 FIELD #1,TT AS X$(X),LR(J)
AS R$(J):TT=TT+LR(J):X=X+1 3050
NEXT 3060 FIELD #1,LT AS TOTAL$:'
facultatif
```

Saisie de nouvelles fiches.

C'est ultra simple. Nous avons vu comment connaître le nombre de fiches NF sitôt après l'OPEN.

```
6310 LINE INPUT "Nom : ",N$:IF
N$="Q" THEN 6360 6320 LSET NOM$=N$
6330 LINE INPUT "Prénom :
",P$:LSET PREN$=P$ 6340 INPUT
"Date en AAMMJJ : ",D$:LSET
DAT$=D$ 6350 PUT
#1,NF+1:NF=NF+1:GOTO 6310 6360
```

```
CLOSE #1 (ou RETURN vers le menu principal)
```

Face à la technique des fichiers "séquentiels", constatez ici la simplicité de ces quelques lignes qui assurent la saisie, le calibrage en longueur maxi, le comptage et la SAUVEGARDE PERMANENTE et AUTOMATIQUE.

Recherche d'une fiche.

Alors ça c'est nettement moins drôle ! En effet pour "sortir" une fiche il faut connaître son numéro... La solution la plus simple consiste à faire des GET de 1 à NF jusqu'à trouver la concordance. C'est ce que nous faisons dans notre programme exemple FICHDIR ; mais ce serait long si le fichier était énorme, à cause de tous ces accès disque. On peut accélérer cela.

Reprenons notre fichier "TRUC" avec ses variables C1$ à C7$. Déterminons quel est le CHAMP CLEF pour la recherche de fiches. Choisissons C3$ et mettons ces valeurs en tableau DIM

```
DIM CLE$(NF) FOR N=1 TO NF:GET
#2,N:CLE$(N)=C3$:NEXT
```

Cela va prendre quelques secondes mais on ne le fait qu'une fois au début. Maintenant les recherches de fiches :

```
410 LINE INPUT "Mot : ",M$ 420 FOR
N=1 TO NF 430 IF
INSTR(CLE$(N),M$)<>1 THEN 460 440
GET #2,N:PRINT C1$, C2$, C3$, C4$,
C5$, C6$, C7$ 450 N=NF  :' arrêt
des recherches 460 NEXT
```

Et c'est de l'ultra rapide ; disons une seconde pour explorer 500 fiches ! Ce INSTR de la ligne 430 surprend un peu : Pourquoi pas IF CLE$(N)<>M$? Parce que les mots dans CLE$(N) sont "complétés" par des blancs, ils proviennent de variables FIELD...

Recours au disque virtuel.

Si vous n'avez pas de disque dur vous allez être pénalisé par la lenteur (relative) du lecteur de disquettes : La méthode intermédiaire consiste à créer un DISQUE VIRTUEL C: en mémoire RAM dans lequel sera copié le gros fichier, et les GET de 1 à NF seront alors ultra rapides puisqu'il y a plus d'inertie mécanique. L'ennui est que ce disque RAM ne peut être créé que par le CONFIG.SYS lors du boutage ; exemple pour obtenir 256 Ko (syntaxe du DOS AMSTRAD) insérer DEVICE=RAMDRIVE .SYS 256 L'AUTOEXEC de votre disquette deviendra alors COPY A:TRUC.FIC C: (copie en disque virtuel C:) GWBasic PROG.BAS (mettra à jour C:TRUC.FIC) COPY C:TRUC.FIC A: (après sortie du Basic par SYSTEM) DEL C:TRUC.FIC

A noter qu'un PC ne possédant qu'un seul drive est suffisant !

Tris de fichiers.

Vous conserverez les méthodes de tris sur tableaux DIM dont vous avez l'habitude, mais ici c'est bien plus simple. Soit à permuter les fiches N et N+1 :

```
GET #1,N:A$=TOTAL$:GET #1,N+1 : '
N mis en A$, N+1 en TOTAL$ PUT
#1,N : ' fiche N+1 en N LSET
TOTAL$=A$:PUT #1,N+1 : ' fiche N
en N+1
```

Remarquez que grâce à la variable globale TOTAL$ il n'est plus nécessaire de programmer un FOR NEXT pour les contenus des diverses rubriques de chaque fiche.

Le programme type. (FICHDIR.BAS)

C'est un logiciel de gestion de fichier en accès direct assez spécial car il n'a pas l'option "création". Un genre cher à l'auteur où tout est déclaré dans le listing même, et sur quelques lignes seulement : En les modifiant on crée une autre application. En somme des "couples" programme-fichier.

En outre ce petit programme est conçu en "structuré", c'est à dire que vous pourrez très facilement le compléter, le gadgétiser à loisir. Il n'y a que les fonctions et sécurités vraiment essentielles, obligatoires. En somme c'est un "KIT de base" solide mais encore petit.

Tout au long du listing, bourré de REM, vous reconnaîtrez les techniques décrites dans cet article, plus bien d'autres dont nous avons déjà parlé dans des numéros précédents de "Compatibles PC Magazine".

Les détails du listing : - tout est en "structuré" c'est à dire en petits modules appelés par des GOSUB. D'où une grande souplesse pour les modifications ultérieures. C'est compilable avec TURBO Basic. - les noms de variables commençant par F sont des "flags", des témoins ou mouchards. Ils valent 0 ou 1 - Lignes 1000-1050 : c'est le menu-carrefour. Pas d'option "chargement", le fichier est déjà ouvert et son nombre de fiches apparait à l'écran. - Ligne 5050 : le contenu de la fiche à effacer est affiché pour accord. Une sécurité pour étourdis. - Ligne 5060 : avantage de cette variable FIELD TOTAL$ qui permet de mettre à blanc une fiche d'un seul coup. - Ligne 22030 : le fait de répondre "Q" sur le premier champ provoque l'abandon de la saisie ou de la modification. (pas de PUT final) - Ligne 22070 : en fin de saisie on demande l'accord pour valider la fiche, car elle sera aussitôt sauvegardée. - Ligne 30040 : la variable talon X$ est sans cesse redéfinie en FIELD dans cette boucle FOR NEXT. Cela peut paraître lourd mais le nombre de champs NRU peut alors être quelconque. - Les sous-programmes utilitaires 49000, 50000 et 51000 sont déjà de vieilles connaissances : affichage et gestion de menus ou de réponses. A noter que les noms de champs (ligne 310) et les noms des options du menu principal (ligne 1020) doivent présenter des lettres initiales différentes.

Mutation SEQUENTIEL-ACCES DIRECT.

Vous exploitez peut-être un fichier "en séquentiel" qui une fois chargé en DIM est devenu à présent fort encombrant. Bref le blocage est pour la prochaine saisie. Vous voudriez le gérer par notre programme mais vous ne voulez pas retaper tout le fichier ! Comme je vous comprends...

Rassurez-vous, rien de plus simple que de transformer votre fichier séquentiel en accès direct :

1 - modifier les lignes de "ACCEDIR" comme si le fichier existait déjà (sous un autre nom que l'actuel)
2 - créez cette ligne provisoire

```
909 GOSUB 40000:END
```

3 - insérez ce sous programme en 40000, lui aussi provisoire.

```
40000 ' copie de séquentiel en
accès direct 40010 OPEN "ACTUEL"
FOR INPUT AS #2:N=0 40020 WHILE
NOT EOF(1):N=N+1 40030 FOR R=1 TO
NRU 40040 LINE INPUT #2,A$:LSET
R$(R)=A$:NEXT 40050 PUT #1,N 40060
WEND:CLOSE 40070 RETURN
```

4 - effacez cette ligne 909, et RUN pour vérifier. Si tout est OK faites DELETE 40000-40070. Et voila le travail.

Bidouille en accès direct.

Il arrive que l'on ait à traiter, modifier un fichier monobloc (binaire ou ASCII) provenant d'un logiciel quelconque. Par exemple remplacer tel "caractère" par tel autre, transformer en majuscules, etc... Il suffit de le considérer comme étant un fichier en accès direct, de le lire par des GET #1 en tronçons de longueur quelconque, puis après traitement, constitution d'un "fichier objet" par des PUT #2.

Autre cas, construire un fichier en accès direct à partir d'un fichier en codage spécial. Un exemple réel, le contenu d'un tableau LOTUS (.WK1) de largeur 1250 (196 lignes sur 102 colonnes !) devait être exploité par un logiciel de mailing conditionnel en Basic : a - une "macro instruction" LOTUS découpe le tableau en six "bandes" verticales, puis les sauvegardes en ASCII soit six fichiers .PRN (cette possibilité LOTUS est limitée à une largeur de 240). b - un programme Basic lit en séquentiel ces fichiers PRN, loge leurs contenus dans les 102 "variables FIELD", et PUT... En final un fichier en accès direct de 245 Ko directement interrogeable par le programme de mailing.

Dans ce genre de bidouilles que l'on appelle des "programmes moulinettes" la solution fait appel une fois sur deux à la technique des fichiers en accès direct.

Conclusion.

Cette technique libère le Basic de ses 60 kilo octets de RAM en lui permettant d'exploiter des fichiers énormes, car ils n'ont pas à être chargés en mémoire. Il faut s'entourer de précautions en programmant la saisie, les modifications ou les effacements, sinon la moindre bêtise sera immédiatement et irrémédiablement sauvegardée ! Si les fichiers en accès direct ont une syntaxe facile à apprendre, il est obligatoire en revanche de bien comprendre leur fonctionnement et leurs "lois" avant de les utiliser...

Michel Archambault

```
10 ' FICHDIR - Gestion de FICHIER en ACCES DIRECT - Michel ARCHAMBAULT 9/4/89
20 ' adaptable en modifiant les lignes 60, 210, 310 et 410
30 ' adjonction d'options en modifiant les lignes 1020, 1030 et 1040
50 DEFINT A-Z:OPTION BASE 0:KEY OFF
60 NRU=3:' nombre de rubriques
100 DIM R$(NRU),LR(NRU),L$(NRU)
110 'R$=noms des champs ; LR=longueurs ; L$=libellés
200 'LONGUEURS
210 DATA 20,15,6
250 LT=0:RESTORE 210:FOR J=1 TO NRU:READ LR(J):LT=LT+LR(J):NEXT
300 'LIBELLES
310 DATA NOM,Prénom,"Date AAMMJJ"
350 RESTORE 310:FOR J=1 TO NRU:READ L$(J):NEXT
400 ' FICHIER DE TRAVAIL
410 FIC$="CLUB":TITRE$="GESTION de FICHIER en ACCES DIRECT"
420 GOSUB 30000:'ouverture de FIC$ en #1
430 NF=LOF(1)/LT:' nombre de fiches
1000 'MENU
1010 CLS:LOCATE 4,2:PRINT"Fichier ";FIC$;" :";NF;"fiches"
1020 DATA SAISIE,CONSULTATION,MODIFICATION,EFFACEMENT,QUITTER
1030 RESTORE 1020:NOPT=5:TIM$=TITRE$:GOSUB 49000
1040 ON K GOSUB 2000,3000,4000,5000,1100
1050 GOTO 1000
1100 CLS:CLOSE:KEY ON
1110 END
2000 'SAISIE de Fiches nouvelles
2010 N=NF+1:FQ=0:' flag quitter
2020 GOSUB 20000:LOCATE 2,30:PRINT"SAISIE (Q pour Quitter)"
2030 GOSUB 22000
2040 IF FQ=1 THEN NF=N-1:FQ=0:GOTO 2060
2050 NF=N:GOTO 2010
2060 RETURN
3000 'CONSULTATION-RECHERCHE
3010 DATA "VISION d'une Fiche N° n",RECHERCHE,Quitter
3020 CLS:TIM$="":NOPT=3:RESTORE 3010:GOSUB 49000
3030 ON K GOTO 3100,3200,3300
3100 CLS:LOCATE 12,28:LINE INPUT"N° de Fiche à visionner ",N$
3110 N=VAL(N$):IF N=0 OR N>NF THEN BEEP:GOTO 3100
3120 GET #1,N:GOSUB 20000:GOSUB 21000
3130 LOCATE 23,36:INPUT"TAPEZ ENTER ",R$:GOTO 3000
3200 CLS:TIM$="Recherche par quelle Rubrique ?":RESTORE 310:NOPT=NRU:GOSUB 49000
3210 R=K:CLS:LOCATE 12,2:PRINT"Valeur recherchée dans ";L$(R);" : ";:LINE INPUT"",V$
3220 LOCATE 15,35:PRINT"Patience..."
3230 FOR N=1 TO NF:GET #1,N
3240 IF INSTR(R$(R),V$)=0 THEN 3260
3250 GOSUB 20000:GOSUB 21000:LOCATE 23,36:INPUT"TAPEZ ENTER ",R$
3260 NEXT
3270 GOTO 3000
3300 RETURN
```

```
4000 ' MODIFICATION
4010 FMOD=1:FQ=0
4020 CLS:LOCATE 12,16:LINE INPUT"N° de la Fiche à MODIFIER (Q pour Quitter) ",N$
4030 IF N$="Q" OR N$="q" THEN 4070
4040 N=VAL(N$):IF N=0 OR N>NF THEN BEEP:GOTO 4020
4050 GET #1,N:GOSUB 20000:GOSUB 21000:LOCATE 2,27:PRINT"MODIFICATION (Q pour
Quitter)":GOSUB 22000
4060 GOTO 4010
4070 FMOD=0:FQ=0
4080 RETURN
5000 ' EFFACEMENT (mise à blanc) d'une Fiche
5010 CLS:LOCATE 12,19:LINE INPUT"N° de la Fiche à EFFACER (Q pour Quitter) ",N$
5020 IF N$="Q" OR N$="q" THEN 5080
5030 N=VAL(N$):IF N=0 OR N>NF THEN BEEP:GOTO 5010
5040 GET #1,N:GOSUB 20000:GOSUB 21000
5050 LOCATE 2,36:PRINT"A EFFACER ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 5010
5060 LSET TOTAL$="":PUT #1,N
5070 GOTO 5010
5080 RETURN
20000 ' AFFICHE LIBELLES
20010 CLS:LOCATE 2,2:PRINT"Fiche N°";N
20020 FOR J=1 TO NRU:LOCATE J+3,2:PRINT L$(J);" ";STRING$(16-
LEN(L$(J)),".");TAB(20+LR(J));">":NEXT
20030 RETURN
21000 ' AFFICHE VALEURS
21010 FOR J=1 TO NRU:LOCATE J+3,20:PRINT R$(J):NEXT
21020 RETURN
22000 ' ENTREES DE VALEURS - FMOD=flag modification
22010 FOR J=1 TO NRU
22020 LOCATE J+3,20:LINE INPUT "",A$
22030 IF J=1 AND (A$="Q" OR A$="q") THEN FQ=1:J=NRU:GOTO 22060
22040 IF FMOD=1 AND A$="" THEN 22060
22050 LSET R$(J)=A$
22060 NEXT:IF FQ=1 THEN 22090
22070 LOCATE NRU+5,38:PRINT"OK ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 22010
22080 PUT #1,N
22090 RETURN
30000 'OUVERTURE DU FICHIER EN ACCES DIRECT
30010 OPEN FIC$ AS #1 LEN=LT
30020 FIELD #1,LR(1) AS R$(1):TT=LR(1)
30030 FOR J=2 TO NRU
30040 FIELD #1,TT AS X$,LR(J) AS R$(J):TT=TT+LR(J)
30050 NEXT
30060 FIELD #1,LT AS TOTAL$
30070 RETURN
49000 ' AFFMENUA = AFFICHAGE de MENU
49010 ' NOPT=Nombre d'Options:TIM$=TITRE du MENU
49020 LOCATE 2,(80-LEN(TIM$))/2:PRINT TIM$
49030 TEX$="":FOR J=1 TO NOPT
49040 READ OPT$:TEX$=TEX$+LEFT$(OPT$,1)
49050 LOCATE J*2+12-NOPT,30:PRINT LEFT$(OPT$,1);" - ";OPT$
49060 NEXT
49070 GOSUB 50000
49080 RETURN
50000 ' MENUA - REPONSE A UN MENU
50010 LM=LEN(TEX$)
50020 LOCATE 24,(70-LM*2)/2:PRINT"Réponse (";
50030 FOR I= 1 TO LM-1:PRINT MID$(TEX$,I,1);",";:NEXT
50040 PRINT RIGHT$(TEX$,1);")";
50050 R$="":WHILE R$="":R$=INPUT$(1):WEND
50060 UPP$=R$:GOSUB 51000:R$=UPP$
50070 K=INSTR(TEX$,R$)
50080 IF K=0 THEN BEEP:GOTO 50050
50090 LOCATE 24,1:PRINT SPACE$(79);
50100 RETURN
51000 ' UPPERA - MISE EN MAJUSCULES
51010 FOR I=1 TO LEN(UPP$)
51020 IF ASC(MID$(UPP$,I,1))>96 AND ASC(MID$(UPP$,I,1))<123 THEN
MID$(UPP$,I,1)=CHR$(ASC(MID$(UPP$,I,1))-32)
51030 NEXT
51040 RETURN
```

DBASE

UN BROWSE DANS UNE FENETRE

"Browse", une instruction de dBase bien agréable pour voir et manipuler les enregistrement d'une base de données.

Malheureusement, Browse travaille en mode plein écran : dans un programme, cela peut être agréable de le mettre dans une fenêtre. En réduisant ses fonctions on peut en faire une procédure pour guider le choix d'un utilisateur, bref un complément indispensable à votre boîte à outils. Aussi nous vous proposons ici une procédure émulant les fonctions de "Browse" à l'intérieur d'une fenêtre.

Un autre avantage d'une telle procédure concerne le travail avec un compilateur dBase. En effet, ceux-ci, pour des raisons de sécurité (sur un réseau par exemple), peuvent refuser de générer le code correspondant à cette instruction.

Vous trouverez accompagnant cet article le listing du fichier "Balaye.Prg" contenant la procédure "Balaye". Nous avons choisi d'appeler ainsi notre procédure car "Browse" est un mot anglais signifiant entre autre parcourir, feuilleter ou butiner ; le mot français Balayer peut lui aussi contenir le concept de parcourir.

Vous pouvez utiliser "Balaye.Prg" pour l'inclure dans un fichier de procédures, ou comme nous l'avons fait pour créer un fichier de procédures (par exemple avec la commande : "MODIFY COMMAND Balaye").

Une fois le texte de ce listing saisi, activez ce fichier de procédure par la commande : "SET PROCEDURE TO Balaye". Vous pouvez alors appeler la procédure "Balaye" avec la commande suivante :

. DO BALAYE WITH Lig0, Col0, Lig9, Col9, "Clef"

Le "WITH" ajouté après le BALAYE permet de passer 5 paramètres à la procédure colonne de départ, puis celles d'arrivée, un peu à la manière d'une commande "@ Lig0, Col0, TO Lig9, Col9".

Le dernier paramètre "Clef" est une chaîne de caractères contenant la formule de calcul de la clef si votre fichier est indexé. Ceci permettra à "BALAYE" de vérifier simplement, après l'édition d'un champ et lors des changement de ligne, si l'ordre d'affichage risque d'être perturbé ou non. Si par contre votre fichier ne possède pas d'index, passez une constante comme dernier paramètre.

```
* Fichier Balaye.Prg : Procédure BALAYE émulant un BROWSE
* dans une fenêtre     Jean-Paul Pruniaux le 18 avril 1989
*
* DO Balaye WITH Ligne Min, Colonne Min, Ligne Max, ;
*                Colonne Max, Expression de surveillance ;
*                des clefs
*
* Les paramètres Ligne / Colonne définissent une fenêtre
* d'affichage (Hauteur mini : 5, largeur mini 3)
*
* L'expression de surveillance des clefs est une chaîne
* de caractères contenant la formule de calcul de la clef
* en cours d'utilsiation (Utiliser une constante si pas de
* clef, par exemple "1" ou "'Constante'")

PROCEDURE Balaye
PARAMETER Lg0, Cl0, Lg9, Cl9, Cle

   @ Lg0, Cl0 TO Lg9, Cl9        && Cadre de la fenêtre
   Lg0 = Lg0 + 2                 && Coordonnées disponibles
   Cl0 = Cl0 + 1                 && pour l'affichage
   Lg9 = Lg9 - 1
   Cl9 = Cl9 - 1

   IF EOF()
      GO TOP
   ENDIF
   PreRec = RECNO()              && Premier record affiché
   MemRec = PreRec               && Record en cours d'édition
   DerRec = 0                    && Dernier record (inconnu)
   MemLig = Lg0                  && Ligne en cours édition
   DerLig = 0                    && Dernière ligne (inconnue)
   PreFld = 1                    && Premier champ affiché
   MemFld = 1                    && Champs en cours édition
   DerFld = 0                    && Dernier champ (inconnu)
   ColFld = 0                    && Ancien PreFld lié à PreCol
   MemCol = Cl0                  && Colonne en cours édition
   DO Colonn                     && Affiche Entete

   AddFic = .F.
   IF EOF()                      && Si fichier vide demander
      DO GetAdd                  && si on ajoute des fiches
      IF .NOT. AddFic
         RETURN
      ENDIF
   ENDIF

   Reorg = .F.                   && Sera vrai pour demander le
                                 && réaffichage après modif
   DO VisLst                     && Affichage de la liste

   DO WHILE .T.                  && Boucle d'édition
      OldCle = &Cle              && Clef avant édition
      DO GetLig                  && Edition d'un champ
      READ
      IF (OldCle # &Cle)         && Changement de l'expression
         Reorg  = .T.            && clef (il faudra réafficher
```

```
      ENDIF                          && la liste au prochain mouvt

      Touche = READKEY()             && Code de la touche de sortie
      IF Touche > 255                && de READ (ramené entre 0 et
         Touche = Touche - 256       && 255)
      ENDIF

      DO CASE                        && AIGUILLAGE PRINCIPAL
         CASE Touche = 0             && Flèche gauche
              DO RecGau

         CASE Touche = 1             && Flèche droite
              DO RecDro
         CASE Touche = 2             && Home
              DO RecGau

         CASE Touche = 3             && End
              DO RecDro

         CASE Touche = 4             && Flèche haute
              DO LigHau

         CASE Touche = 5             && Flèche basse
              DO LigBas

         CASE Touche = 6             && Page Up
              DO PagHau

         CASE Touche = 7             && Page Down
              DO PagBas

         CASE Touche = 12            && Escape
              EXIT

         CASE Touche = 14            && Ctrl End
              EXIT

         CASE Touche = 15            && Champ rempli, -> suivant
              DO RecDro

         CASE Touche = 16            && Entrée
              DO RecDro

         CASE Touche = 33            && Ctrl-Home

         CASE Touche = 34            && Ctrl-PgUp
              GO TOP
              PreRec = RECNO()          && Premier record affiché
              MemRec = PreRec           && Champ en cours édition
              MemLig = Lg0              && Ligne en cours édition
              DO VisLst

         CASE Touche = 35            && Ctrl-PgDn
              GO BOTTOM
              PreRec = RECNO()          && Premier record affiché
              MemRec = PreRec           && Record en cours d'édition
              MemLig = Lg0              && Ligne en cours édition
              DO VisLst
         CASE Touche = 36            && F1 (Aide)

      ENDCASE
   ENDDO
RETURN
*——              && Calcul la colonne d'affichage du champ
PROCEDURE Colonn  && en cours, affiche éventuelt les titres
                  && ————————————————————————

   TstVis = (ColFld # PreFld)  && Vrai si titre à afficher
   IF TstVis
      L = Lg0 -1                  && Efface ligne de titre
      @ L, C10 CLEAR TO L,C19
      ColFld = PreFld
   ENDIF

   Fld = PreFld                   && Boucle du premier champs
```

```
. * Exemple de fichier non indexé
. USE Clients
. SET PROCEDURE TO Balaye
. DO Balaye WITH 3,10, 20,50,"1"
. * Exemple avec un fichier index
. INDEX ON UPPER(Nom) TO Clients
. DO Balaye WITH 3,10, 20,50,
"UPPER(Nom)"
```

Une fois dans "Balaye", la fenêtre se dessine, la première ligne affiche les noms des enregistrements, et dessous viennent autant d'enregistrements que la fenêtre peut en contenir. La dernière ligne reste vierge pour pouvoir afficher un message.
Une ligne est alors en vidéo inversée et le curseur est positionné sur l'un des caractères de cette ligne. Vous pouvez alors modifier le champ mis ainsi en évidence.
Les flèches (y inclus "Page ver le haut" et "Page vers le bas") vous permettent de vous déplacer d'un champs à l'autre et d'une ligne à une autre.
Si le champ le plus à droite est plus large que la fenêtre ne le permet, seuls les premiers caractères sont affichés. Cependant, lorsque vous en serez à modifier ce champs, vous pourrez en faire défiler le contenu grâce aux flèches droite et gauche pour voir ou modifier le texte situé au delà de la dernière colonne.
Si vous modifiez l'expression clef, la fenêtre sera réorganisée, et affichée de nouveau lors du prochain changement de ligne.
Vous sortez de "Balaye" soit par la touche "Escape" ou "Echappement" soit par les touches "Contrôle Fin" ou "Contrôle W".
Ce programme a été écrit pour "dBase III+" (et tout produit 100% compatible).

CE QUI REND POSSIBLE "BALAYE".

La fonction majeure permettant l'écriture d'un programme comme "Balaye" est la fonction READKEY().
Après une séquence "@ ... GET ...", "READ" permet de connaître la dernière touche qui a été frappée par l'utilisateur avant de quitter l'édition.
READKEY() retourne un code permettant d'identifier cette touche, celui-ci est supérieur à 255 si la donnée en cours d'édition a été modifiée, ou un code inférieur à 255 si cette donnée n'a pas été modifiée. Les codes retournés par READKEY() sont les suivants :
0 ou 256 : Effacement arrière ou flèche gauche 1 ou 257 : Flèche droite 2 ou 258 : Home (recul d'un mot) 3 ou 259 : End (avance d'un mot) 4 ou 260 : Flèche vers le haut (champ précédent) 5 ou 261 : Flèche vers le bas (champ suivant) 6 ou 262 : Page précédente 7 ou 263 : Page suivante 12 : Escape ou Echappement (Abandon) 14 ou 270 : Contrôle Fin (Sauvegarde) 15 ou 271 : Champs remplis, curseur arrivé au bord droit 16 ou

272 : Entrée (validation du champ) 33 ou 289 : Contrôle Home (habituellement utilisé pour appeler un champ mémo) 34 ou 290 : Contrôle Page Vers le Haut 35 ou 291 : Contrôle Page Vers le Bas 36 ou 292 : F1 : Touche de fonction associée à l'aide.
Ainsi après la saisie d'un champ, nous pourrons tester le code en retour de cette fonction pour savoir si nous devons éditer le champ suivant, le champ précédant, remonter d'une ligne, ...
Une autre fonctionnalité importante de "dBase" que nous avons utilisée réside dans les macros. Le symbole "&" suivi d'un nom de variable de type chaîne de caractères permet de remplacer un morceau de commande ou de source de programme par le contenu de cette variable :

. Var = "UPPER (Nom)" . INDEX ON &Var TO Client

Deviendra à l'exécution

. INDEX ON UPPER (Nom) TO Client

Ceci nous est utile à plusieurs endroits de "Balaye". Par exemple lorsque nous surveillons les changements de valeur de la clef, nous pouvons mémoriser la clef avant la modification d'un champ par la ligne :

. OldCle = &Cle

Ce qui deviendra à l'exécution :

. OldCle = UPPER (Nom)

Ceci revient à dire que nous sommes capables de modifier le programme en fonction du contenu de certaines variables.
Une dernière fonctionnalité importante de "dBase" que nous avons utilisé est dans la fonction FIELD(Numero). Celle-ci nous retourne une chaîne de caractères précisant le nom du champ de numéro donné. Ainsi si la base de données comprend les champs "Nom, Prenom, Adresse ...", FIELD(1) renvoie "NOM", et "FIELD(3)" renvoie "ADRESSE".

Ceci nous est utile, grâce à une macro "&" pour visualiser ou éditer le contenu d'un champ :

. NomChamp = FIELD (3) . @ 10, 10 GET &NomChamp

Devient à l'exécution :

. @ 10, 10 GET ADRESSE

Si le numéro transmis à FIELD() est trop grand, cette fonction retourne une chaîne

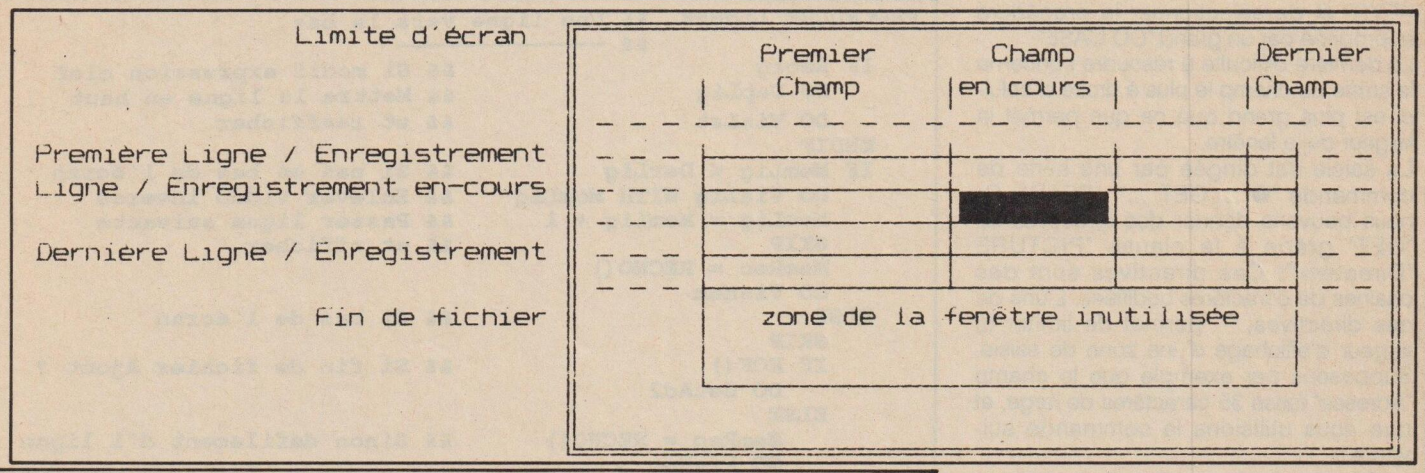
```
   F    = FIELD (Fld)              && à afficher au dernier
   Col = Cl0
   DO WHILE LEN(F) # 0 .AND. Col<Cl9
      IF Fld = MemFld
         MemCol = Col             && Colonne du champs en cours
      ENDIF
      Col2= Col + LEN (&F)
      IF TstVis                   && Affichage titre
         Vis = F
         IF Col2 > Cl9            && Vérifie l'encombrement
            Vis = LEFT (Vis, Cl9 - Col + 1)
         ENDIF
         IF LEN (Vis) > LEN (&F)
            Vis = LEFT (Vis, LEN(&F))
         ENDIF
         @ L, Col SAY Vis
      ENDIF
      Col = Col2 + 1              && Paasage colonne suivante,
      Fld = Fld + 1               && champ suivant
      F   = FIELD (Fld)
   ENDDO
RETURN
*——
PROCEDURE VisLig  && Visualise une ligne «Lig» de la fenêtre
PARAMETER Lig     && ————————————————————
   DerFld = PreFld                && Boucle sur les champs
   Fld    = FIELD (DerFld)        && (recalcul DerFld)
   Col    = Cl0                   && et sur les colonnes)
   DO WHILE LEN(Fld) # 0 .AND. Col<Cl9
      Vis = &Fld
      Col2 = Col + LEN (Vis)      && Vérifie la taille de la
      IF Col2 > Cl9               && fenêtre
         Vis = LEFT (Vis, Cl9 - Col + 1)
      ENDIF
      @ Lig, Col SAY Vis

      Col = Col2+1                && Colonne suivante
      DerFld = DerFld+1           && Champ suivant
      Fld    = FIELD (DerFld)
   ENDDO
   DerFld = DerFld - 1            && Mise à jour dernier champ
RETURN
*——
PROCEDURE VisMem  && Visu en vidéo inverse la ligne en cours
                  && ————————————————————
   GO MemRec
   SET COLOR TO N/W
   DO VisLig WITH MemLig
   SET COLOR TO
RETURN
*——
PROCEDURE GetLig  && Saisie du champ en cours
                  && ————————————————
   Fld = FIELD (MemFld)
   Cl2 = MemCol + LEN (&Fld)      && Colonne de fin
   IF Cl2 > Cl9                   && si supérieur à colonne max
                                  && utiliser PICTURE "@Sxxx"
                                  && pour gérer un défilement
      NbrCar = Cl9 - MemCol + 1
      Pct    = "@S"+LTRIM(STR(NbrCar))
      @ MemLig, MemCol GET &Fld PICTURE '&Pct'
   ELSE
      @ MemLig, MemCol GET &Fld
   ENDIF
RETURN
*——
PROCEDURE VisLst  && Visualise tous les enreg.
                  && ————————————————
   @ Lg0, Cl0 CLEAR TO Lg9,Cl9 && Efface fenêtre

   DerLig = Lg0                   && Boucle avec recalcul
   GO PreRec                      && de la «Dernière ligne»
   DO WHILE (DerLig < Lg9) .AND. .NOT. EOF()
      DO VisLig WITH DerLig
      DerLig = DerLig+1
```

```
      SKIP
   ENDDO
   DerLig = DerLig - 1
   SKIP -1
   DerRec = RECNO()                && Mise à jour paramètres
   DO VisMem
RETURN
*——
PROCEDURE PagBas   && Une page (un écran) vers le bas
                   && ————————————————
   IF Reorg                        && Pas faite si modif de
      DO DepLig                    && l'expression clef
   ELSE
      GO DerRec                    && Enregistrement du bas
      SKIP                         && Suivant
      IF EOF()
         DO GetAd2                 && Si fin demander si ajout
         RETURN
      ENDIF
      PreRec = RECNO()             && Mise à jour paramètre
      MemRec = PreRec
      MemLig = Lg0
   ENDIF
   DO VisLst                       && Réaffichage
RETURN
*——
PROCEDURE PagHau   && Une page (un écran) vers le haut
                   && ————————————————
   IF Reorg                        && Si modif expression clef
```

de longueur nulle. Ceci permet de tester l'existence d'un champ de numéro donné, ou encore de savoir si on a exploité tous les champs de la base de données.

COMMENT EST ORGANISE "BALAYE".

Les éléments manipulés par "Balaye" sont les suivants :
- Une fenêtre contenant entre autre une première ligne, une ligne en vidéo-inverse et une dernière ligne. - A chaque ligne est associé un enregistrement de la base de données, donc à la première ligne un premier enregistrement, de même un enregistrement pour la ligne en vidéo inversée et un pour la dernière ligne. - Sur une ligne peuvent être visualisés plusieurs champs d'un même enregistrement. Entre autre un premier champ, un champ en cours d'édition et un dernier champ.
Ceci peut se visualiser dans le schéma ci dessous :




```
      DO DepLig                    && Paramétrer la ligne en
      SKIP                         && cours comme début d'écran
   ELSE
      GO PreRec                    && Sinon repositioner en haut
   ENDIF
   SKIP Lg0-Lg9                    && Recule dans le fichier de
   PreRec = RECNO()                && la hauteur de l'écran
   MemRec = PreRec
   MemLig = Lg0
   DO VisLst
RETURN
*——
PROCEDURE RecDro   && Un champ (record) à droite
                   && ————————————————
   Action = .F.                    && Vrai si raffraichissage de
                                   && l'affichage
   IF MemFld >= DerFld             && Teste si bord droit
      IF LEN (FIELD (PreFld + 1)) > 0 && et si dernier champ
         PreFld = PreFld + 1
         IF Reorg
            DO DepLig
         ENDIF
         DO VisLst
         Action = .T.
      ENDIF
   ENDIF
   IF MemFld < DerFld              && Teste s'il reste un champ
```

Ce schéma illustre les variables clefs que nous gérons dans "Balaye". "PreRec", "MemRec" et "DerRec" sont des numéros d'enregistrement, respectivement celui de la première ligne, de la ligne en cours d'édition (enregistrement mémorisé), et celui de la dernière ligne ; de la même manière "PreFld", "MemFld" et "DerFld" mémorisent des numéros de champs. "MemCol" sert à mémoriser la colonne d'écran ou commence l'affichage du champ correspondant à "MemFld".
La mémorisation de ces informations va permettre de gérer simplement les mouvements de la zone en vidéo inversée, ou les mises à jour d'affichage dans la fenêtre.
Par exemple si l'utilisateur de notre programme frappe la touche "Flèche Vers le Bas" pour terminer l'édition du champs en cours ("MemFld"), nous pouvons comparer "MemLig" avec "DerLig" pour savoir s'il existe un autre enregistrement affiché en dessous. Si cet enregistrement existe, il suffit de le mettre en vidéo inversée (en

pensant à avancer dans le fichier, et en pensant à mettre à jour les valeurs "MemLig" et "MemRec").

Si par contre nous sommes sur la dernière ligne, nous pouvons nous repositionner sur le premier enregistrement, avancer d'un enregistrement pour être sur le deuxième, le déclarer comme étant le premier et refaire un affichage général. Ceci va se traduire par le défilement d'un enregistrement vers le haut à l'écran.

De la même manière, nous pouvons gérer le déplacement latéral d'un champ à un autre en comparant "MemFld" avec "PreFld" ou "DerFld".

Grâce à ceci nous pouvons écrire une série de petites procédures gérant : Le déplacement d'un champ vers la droite, celui d'un champ vers la gauche, celui d'une ligne vers le bas ou vers le haut, et celui d'une page vers le bas ou vers le haut.

Après l'édition du champ en cours, il suffit de récupérer la valeur fournie par "READKEY()" et de sélectionner la procédure appropriée par un grand "DO CASE".

La dernière difficulté à résoudre concerne la saisie du champ le plus à droite si celui-ci est plus grand que ce que permet la largeur de la fenêtre.

La saisie est dirigée par une série de commande "@ ... GET ...", "READ". Or nous pouvons donner des directives au "GET" grâce à la clause "PICTURE "Directive"". Ces directives sont des chaînes de caractères codifiées. L'une de ces directives, "" permet de limiter la largeur d'affichage d'une zone de saisie. Supposons par exemple que le champ "Adresse" fasse 35 caractères de large, et que nous utilisions la commande suivante :

@ 10,10 GET Adresse PICTURE " READ

Seuls les 15 premiers caractères de "Adresse" seront visualisés dans un premier temps. Le curseur sera sur le premier de ces caractères. Si nous modifions les 15 premiers caractères, ou si nous déplaçons 15 fois le curseur à droite, nous serons alors sur le dernier caractère. La saisie d'un nouveau caractère ou le déplacement du curseur va faire défiler sur la gauche le contenu de la zone de saisie pour montrer la suite de "Adresse". Et nous pouvons continuer ainsi jusqu'à arriver au dernier caractère. Nous pouvons provoquer le déplacement inverse avec la flèche à gauche.

Avec le listing de la procédure "Balaye", nous vous avons fourni un petit programme "TstBal" permettant de tester celle-ci, et de vous montrer comment utiliser "Balaye". Maintenant, à vous de vous mettre à vos claviers, et de mettre des "BROWSE" dans des fenêtres, ou dans vos programmes compilés.

Jean Paul Pruniaux

```
      MemFld = MemFld + 1
      Action = .T.
   ENDIF
   IF Action                          && Gère l'affichage
      DO Colonn
   ELSE
      DO LigBas
   ENDIF
RETURN
*——
PROCEDURE RecGau   && Un champ (record) à gauche
                   && ———————————————
   IF (MemFld <= PreFld) .AND. (PreFld > 1)
      PreFld = PreFld - 1             && Mouvement à gauche
      MemFld = PreFld
      DO Colonn
      IF Reorg
         DO DepLig
      ENDIF
      DO VisLst
      RETURN
   ENDIF
   IF (MemFld > PreFld)               && Translation sans mouvement
      MemFld = MemFld - 1
      DO Colonn
      RETURN
   ENDIF
   DO LigHau                          && si rien monter d'une ligne
RETURN
*——
PROCEDURE LigBas   && Une ligne vers le bas
                   && ———————————————
   IF Reorg                           && Si modif expression clef
      DO DepLig                       && Mettre la ligne en haut
      DO VisLst                       && et réafficher
   ENDIF
   IF MemLig < DerLig                 && Si pas en bas de l'écran
      DO VisLig WITH MemLig           && Enlever vidéo inverse
      MemLig = MemLig + 1             && Passer ligne suivante
      SKIP                            && et afficher
      MemRec = RECNO()
      DO VisMem
   ELSE                               && Si bas de l'écran
      SKIP
      IF EOF()                        && Si fin de fichier Ajout ?
         DO GetAd2
      ELSE
         MemRec = RECNO()             && Sinon défilement d'1 ligne
         GO PreRec
         SKIP
         PreRec = RECNO()
         DO VisLst
      ENDIF
   ENDIF
RETURN
*——
PROCEDURE LigHau   && Une ligne vers le haut
                   && ———————————————
   IF Reorg                           && Si modif expression clef
      DO DepLig                       && Mise à jour paramètres
      SKIP -1                         && Remonte une ligne
      IF .NOT. BOF()                  && Teste limites
         PreRec = RECNO()
         MemLig = Lg0 +1
         SKIP
      ELSE
         GO MemRec
      ENDIF
      DO VisLst                       && Réaffiche
   ENDIF
   IF MemLig > Lg0                    && Teste si première ligne
      DO VisLig WITH MemLig           && Remontée d'une ligne
      MemLig = MemLig - 1
      SKIP -1
      MemRec = RECNO()
      DO VisMem
```

```
   ELSE                          && En première ligne
      SKIP -1                    && remonte et défile
      IF .NOT. BOF()
         PreRec = RECNO()
         MemRec = PreRec
         DO VisLst
      ELSE
         GO MemRec
         ?? CHR(7)
      ENDIF
   ENDIF

RETURN
*——
PROCEDURE DepLig  && Mise à jour paramètres après modif clef
                  && ————————————————————————
   PreRec = MemRec
   MemLig = Lg0
   Reorg  = .F.
RETURN
*——
PROCEDURE GetAdd  && Traitement fin de fichier : Ajout ?
                  && ————————————————————————
   ?? CHR(7)                     && Beep !
   IF Cl9-Cl0 >= 7               && Si largeur suffisante
      @ Lg9,Cl0 SAY "Ajout" GET AddFic
      READ                       && Question Ajout ?
      @ Lg9,Cl0 SAY "       "
      IF AddFic                  && Si Ok : Ajout
         APPEND BLANK
      ENDIF
   ELSE
      AddFic = .F.
   ENDIF
RETURN
*——
PROCEDURE GetAd2  && Traitement fin de fichier : Ajout ?
                  && Avec gestion de l'affichage fenêtre
                  && ————————————————————————
   DO GetAdd                     && Demande si ajout
   IF AddFic
      Reorg = .T.                && Nouvelle clef
      IF MemLig < Lg9 -1         && Si ligne dispo l'utiliser
         Rec = RECNO()
         Go MemRec
         DO VisLig WITH MemLig
         GO Rec
         MemRec = Rec
         MemLig = DerLig +1
         DO VisMem
      ELSE                       && Pas de ligne dispo
         MemRec = RECNO()        && Régénérer l'affichage
         DO DepLig
         DO VisLst
      ENDIF
   ELSE
      GO MemRec
   ENDIF
RETURN
```

```
* TstBal.Prg : Test de la procédure «Balaye»

SET TALK        OFF
SET STATUS      OFF
SET SCOREBOARD OFF
CLEAR
@ 0,0 SAY "Test de la procédure BALAYE, émulation de BROWSE dans une fenêtre"
@ 1,0 SAY "Jean-Paul Pruniaux le 18 avril 1989, pour Compatibles PC Magazine"

SET PROCEDURE TO BALAYE
USE Clients INDEX Clients

DO Balaye WITH 4, 10,11,70, "UPPER(Nom)"
DO Balaye WITH 12,37,20,43, "UPPER(Nom)"

RETURN
```

PASCAL

DESSINER EN TURBO PASCAL 4

Ce programme permet de créer un dessin, de le modifier et de le sauvegarder dans un fichier. Sa visualisation se fera en mode graphique.

L'encadré reproduit l'unité "Detect" qui sert à détecter quelle graphique est présente et de configurer le mode graphique, selon le cas, en Hercules, CGA, EGA ou VGA.

Louala Nasser

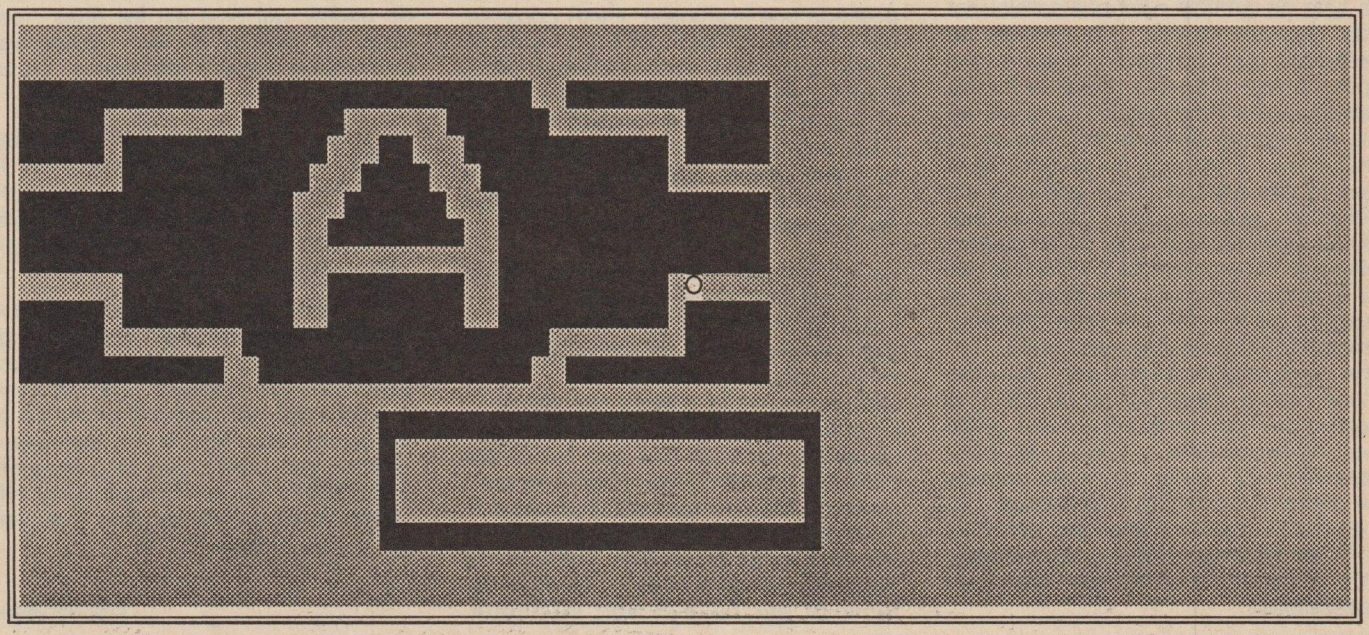
```
program creation_dessin;

(* Utilisation de diverses unités *)

uses crt,dos,graph,detect;

const  Xmax=78; (* Longueur maximale du dessin *)
       Ymax=21; (* Largeur maximale du dessin *)

       Pv=#32;  (* Code ASCII de la touche < Espace > *)
       Esc=#27; (* Code ASCII de la touche < Escape > *)

       Lf=#75;  (* Code ASCII de la touche < Flèche gauche > *)
       Rg=#77;  (* Code ASCII de la touche < Flèche droite > *)
       Up=#72;  (* Code ASCII de la touche < Flèche haut > *)
       Dw=#80;  (* Code ASCII de la touche < Flèche bas > *)

type   chaine=string[Xmax];
       fichier=file of chaine;
       nomfich=string[8];

var    tableau:array[1..Xmax,1..Ymax] of byte;
       choix:char;
       nomf:nomfich;
       xi,yi:byte;

procedure cardillage;

(* Domaine de visualisation du dessin  *)

begin
  clrscr;
  for xi:=1 to Xmax do
  begin
    gotoxy(1+xi,2);write('˜');gotoxy(1+xi,3+Ymax);write('˜');
  end;
  for yi:=1 to Ymax do
  begin
    gotoxy(1,yi+2);write('˜');gotoxy(Xmax+2,yi+2);write('˜');
  end;
  gotoxy(1,2);write(' ');gotoxy(1,Ymax+3);write('¨');
  gotoxy(2+Xmax,2);write('»');gotoxy(Xmax+2,Ymax+3);write('…');
  gotoxy(5,1);
  write('[ Flèches ] : déplacement / [ Esp ] : dessine / [ Esc ] : pour arrêter');
  window(2,3,Xmax+1,Ymax+4);
  for xi:=1 to Xmax do for yi:=1 to Ymax do
```

```
  begin
    gotoxy(xi,yi);write(chr(176));
  end;
end;

procedure deplace(x,y:byte;pave:boolean);

(* Déplace le curseur et laisse ou non suivant la valeur de pavé une trace *)

begin
  gotoxy(xi,yi);
  if not(pave) then begin write(chr(176));tableau[x,y]:=0 end
  else              begin write(chr(219));tableau[x,y]:=1 end;
  gotoxy(x,y);write('O');xi:=x;yi:=y;
end;

procedure lect_dessin;

(*
   Dessine en faisant appel à la procedure deplace :
   Le dessin se fait à l'aide des touches de déplacements, La touche Espace
   sert à visualiser ou non le tracé, la touche <Esc> : fin de l'execution
*)

var  x,y:byte;
     choix:char;
     pave:boolean;

begin
  xi:=40;yi:=10;x:=40;y:=10;pave:=false;deplace(x,y,pave);choix:=readkey;
  repeat
    if choix=Pv                then pave:=not(pave);
    if (choix=Lf) and (x>1)    then dec(x);
    if (choix=Rg) and (x<Xmax) then inc(x);
    if (choix=Up) and (y>1)    then dec(y);
    if (choix=Dw) and (y<Ymax) then inc(y);
    deplace(x,y,pave);choix:=readkey
  until choix=Esc;window(1,1,80,25);
end;

procedure affiche_dessin(nomf:nomfich);

(* Visualise le dessin en mode texte quadrillé *)

var chain:chaine;
    fich:fichier;
begin
  clrscr;assign(fich,nomf);reset(fich);cardillage;yi:=0;
  repeat
    read(fich,chain);inc(yi);
    for xi:=1 to Xmax do
    begin
      gotoxy(xi,yi);
      if copy(chain,xi,1)='1' then
      begin
        tableau[xi,yi]:=1;write(chr(219))
      end
      else begin
            tableau[xi,yi]:=0;write(chr(176))
          end;
    end;
  until eof(fich);close(fich)
end;
```

```
procedure sauvegarde(Yam:byte;nomf:nomfich);

(* Sauvegarde le dessin creé dans un fichier *)

var car:string[1];
    chain:chaine;
    fich:fichier;

begin
  assign(fich,nomf);rewrite(fich);chain:='';
  for yi:=1 to Yam do
  begin
    for xi:=1 to Xmax do
    begin
      str(tableau[xi,yi],car);chain:=chain+car;
    end;write(fich,chain);chain:='';
  end;close(fich);
end;

procedure creer_dessin;

(* Appel la procedure Lect_dessin / Yam = largeur maximale du dessin *)

var Yam:byte;
    nomf:nomfich;
    pave:boolean;

begin
  lect_dessin;
  for yi:=1 to Ymax do
  begin
    pave:=false;xi:=1;
    repeat
      if tableau[xi,yi]=1 then pave:=true;inc(xi)
    until (pave) or (xi>Xmax);
    if pave then Yam:=yi;
  end;
  gotoxy(1,25);write('Sauvegarde du dessin (o)ui ?' : ');choix:=readkey;
  if upcase(choix)='O' then
  begin
    gotoxy(1,25);clreol;gotoxy(1,25);write('Nom du dessin ? : ');
    readln(nomf);sauvegarde(Yam,nomf)
  end;
end;

procedure dessin_Graph(nomf:nomfich);

(* affiche la figure en mode graphique *)

var chain:chaine;
    fich:fichier;

begin
  assign(fich,nomf);reset(fich);detection;ClearDevice;yi:=0;
  repeat
    read(fich,chain);inc(yi);
    for xi:=1 to Xmax do
    begin
      if copy(chain,xi,1)='1' then PutPixel(xi+100,yi+100,1)
      else PutPixel(xi+100,yi+100,0);
    end;
  until eof(fich);
```

```
  close(fich);choix:=readkey;closegraph
end;

(* présentation du menu général et initialisation de tableau *)

begin
  repeat
    for yi:=1 to Ymax do for xi:=1 to Xmax do tableau[xi,yi]:=0;clrscr;
    gotoxy(20,4);write('C) réation d''un dessin');
    gotoxy(20,6);write('L) ecture d''un dessin en mode text');
    gotoxy(20,8);write('M) odification d''un dessin');
    gotoxy(20,10);write('A) ffichage du dessin en mode graphique');
    gotoxy(20,12);write('F) in de travail');
    gotoxy(23,14);write('choix : ');choix:=readkey;write(upcase(choix));
    case upcase(choix) of

      'C':begin
            cardillage;creer_dessin
          end;
      'L':begin
           gotoxy(20,20);write('Nom du dessin: ');
           readln(nomf);affiche_dessin(nomf);choix:=readkey;window(1,1,80,25)
          end;
      'M':begin
            gotoxy(20,20);write('Nom du dessin: ');readln(nomf);
            affiche_dessin(nomf);creer_dessin
          end;
      'A':begin
            gotoxy(20,20);write('Nom du dessin : ');readln(nomf);
            dessin_Graph(nomf)
          end;
    end;
  until upcase(choix)='F';clrscr
end.
```

[Flèches] : déplacement / [Esp] : dessine / [Esc] : pour arrêter



GESTION D'IMPRIMANTE EN TURBO PASCAL

Existe-t-il un moyen de connaître l'état d'une imprimante (présence ou absence, imprimante prête ou non , ...) ?

La réponse à cette question est complexe , en Turbo Pascal il est possible de gérer les erreurs d'exécution et par la même de récupérer les erreurs survenues à l'impression. Il n'existe malheureusement que deux codes d'erreurs renvoyés par la variable IoResult concernant l'imprimante, 159 : détection fin de papier et 160 : erreur d'écriture sur un périphérique. Le contrôle de ces erreurs (répertoriées erreurs critiques dans le manuel de référence de Turbo Pascal) passe obligatoirement par les procédures standard write(lst,xxx) et writeln(lst,xxx).

Une suite d'instruction du type :

```
{$I-}(* désactivation du contrôle
par turbo pascal *)
writeln(lst,xxx); (* écriture sur
l'imprimante *)
{$I+} (* réactivation du contrôle
automatique *)
erreur_acces:=ioresult; (* récupération du
code retour *)
if erreur_acces<>0 then writeln('erreur
imprimante :',erreur_acces);
```

permet de récupérer l'erreur d'exécution si celle-ci se produit.
Ce type de contrôle n'est pas satisfaisant pour plusieurs raisons : - le contrôle ne s'effectue qu'à posteriori, c'est à dire après envoi d'un ou d'une suite de caractères sur l'imprimante. - il existe un cas où le programme reste 'bloqué' en attente de démarrage du périphérique, c'est le cas typique où l'imprimante est alimentée mais non prête (laissant tout lieu de croire à une boucle infinie ou à un 'plantage' du programme). - Dans le cas où l'imprimante n'est pas alimentée le programme se déroule correctement, l'impression s'effectuant dans le vide.
Une autre solution passe par l'utilisation d'une routine d'interruption du BIOS, la gestion de l'imprimante est en effet gérée par l'interruption 17H (H pour hexadécimal) . Cette routine est capable de prendre en compte la présence d'au maximum 3 cartes d'interface parallèle, soit 3 imprimantes différentes, le tableau n°3 donne toutes les fonctions accessibles .

tableau n° 1 (récapitulatif des octets renvoyés par l'interruption 17H) :

	NEC P 2200	EPSON LX 800	OLIVETTI DM 100	OKI microline 294
plus de papier	40	56	144	40
imprimante éteinte	200	192	48	48
ok , paré à imprimer	144	144	144	144
allumé, non prête	8	24	144	0
pas d'imprimante	176	176	48	48

En consultant ce tableau, on constate que cette routine possède 3 fonctions : - la fonction n° 0 dont le travail consiste à effectuer l'impression du caractère contenu dans le registre AL. - la fonction n° 1 qui a la charge de l'initialisation du périphérique de sortie. - la fonction n°2 qui permet de lire l'état de l'imprimante.
Grâce à cette interruption il est possible d'interroger l'imprimante et de récupérer l'état de celle-ci sans avoir besoin d'envoyer un caractère. Le turbo pascal possède une instruction (IntR contenue dans l'unité standard DOS) qui autorise l'exécution d'une routine d'interruption. Dans le cas qui nous intéresse, il s'agit de la routine 17H et de sa fonction n°2 appel . Le résultat de l'exécution est ensuite récupéré dans le registre AH, celui-ci est constitué d'un octet qui représente l'état de l'imprimante.

La signification des différents bits de l'octet est donnée par le tableau n°4.
Le programme INT17H.PAS fourni un exemple d'utilisation de l'interruption 17h via la procédure standard IntR.
Cette solution répond-elle à notre attente ?
L'expérience en informatique aidant, l'auteur a voulu vérifier que les codes retournés correspondaient bien à ce qu'ils devaient être quel que soit l'ordinateur et quel que soit l'imprimante. La logique voudrait que pour une imprimante parallèle compatible PC, l'octet d'état renvoyé par celle-ci soit identique.

Comme le lecteur pourra le constater (tableau n°1), il n'en est rien du moins en ce qui concerne les quatre modèles que l'auteur a pu tester. Selon la marque et le modèle d'imprimante l'octet renvoyé par l'interruption 17H pour un état donné est différent, la seule constante (pour ces 4 modèles) est le code 144 qui indique que l'imprimante existe et qu'elle est prête à imprimer. Il est à noter que parmi tous les modèles testés, il en est un (l'Olivetti DM100) qui ne retourne que deux types de codes, qui plus est les états : prêt, plus de papier et non prête sont considérés comme identiques (code 144 !).

Au vu de ces résultats, il est maintenant facile d'expliquer pourquoi dans les logiciels commerciaux un problème imprimante se traduit par un message aussi peu explicite que : imprimante non prête ou device PRN error.

L'écriture d'une routine permettant de tester l'imprimante devient par conséquent problématique, il parait difficile de réaliser une fonction "universelle" sauf peut être en ce qui concerne l'état 'Ready'; un exemple de gestion est donné par le programme TESTIMP.PAS pour l'imprimante NEC P2200, le lecteur devra donc l'adapter à son propre modèle.

Eric Servagnat

tableau n° 2 (signification des octets renvoyés par l'interruption 17H) :

	bit n° 7	6	5	4	3	2	1	bit n° 0	
8 (8H)	0	0	0	0	1	0	0	0	not ready
24 (18H)	0	0	0	1	1	0	0	0	not ready
40 (28H)	0	0	1	0	1	0	0	0	plus de papier
48 (30H)	0	0	1	1	0	0	0	0	éteinte
56 (38H)	0	0	1	1	1	0	0	0	plus de papier
144 (90H)	1	0	0	1	0	0	0	0	ok
176 (B0H)	1	0	1	1	0	0	0	0	pas d'imprimante
192 (C0H)	1	1	0	0	0	0	0	0	imprimante éteinte
200 (C8H)	1	1	0	0	1	0	0	0	imprimante éteinte
	not busy	acknowledge	out of paper	selected	I O / ERROR		inutilisés	time out	

tableau n° 4 (signification de l'octet d'état) :

bit n°	état
7	1=not busy 0 = busy
6	1=acknowledge
5	1=out of paper
4	1=selected
3	1=I/O ERROR
2	inutilisé
1	inutilisé
0	1=TIME OUT

tableau n°3 (fonctions de l'interruption 17H de gestion de l'imprimante) :

AH = 0	impression du caractère contenu dans AL sur l'imprimante dont le numéro est chargé dans DX (0, 1 ou 2) En sortie : AH = 1 -> le caractère n'a pus être imprimé correctement (causé par un TIME OUT) Ah <> 1 -> code retour de l'état de l'imprimante
AH = 1	initialise la sortie imprimante dont le numéro est chargé dans DX, en sortie obtention de l'état de l'imprimante dans AH
AH = 2	Lecture de l'état de l'imprimante dont le numéro est chargé dans le registre DX, en sortie état de l'imprimante dans AH

```
Program INT17H;
{
Auteur : SERVAGNAT Eric
}

Uses Crt,Dos;


Var
   Regs : Registers;
   Code_retour : byte;
   bidon : char;


{ fonction retournant un chaine contenant le nombre transformé en binaire }

Function DecBin(nombre:integer):string;

var
   chaine : string;
   i : byte;
begin

 chaine:='';
 repeat
   if odd(nombre)
   then chaine:='1'+chaine
   else chaine:='0'+chaine;
   nombre:= nombre div 2;
 until nombre = 0;
 for i:=length(chaine) to 7 do chaine:='0'+chaine;
 decbin:=chaine;

end;  { fin de decbin }


{ procédure effectuant un remise à blanc des registres }

procedure Efface_Reg;
begin
 with regs do
  begin
   ax:=0;bx:=0;cx:=0;dx:=0;bp:=0;si:=0;di:=0;ds:=0;es:=0;
  end;
end;


Function Interroge_Imprimante : byte;
begin
 Efface_Reg;
 Regs.AH:=2;          { appel de la fonction n° 2 }
 IntR($17,Regs);      { éxécution de la routine d'interruption  17H }
 interroge_imprimante:=Regs.AH; { récupération du code d'état }
end;


begin
 writeln('code retour imprimante : ',Interroge_Imprimante:3,' en décimal ');
 writeln;
 writeln('soit en binaire        : ',DecBin(Interroge_Imprimante));
 writeln;
 bidon:=readkey;
end.
```

```
Program TESTIMP;
{
Auteur : SERVAGNAT Eric
}

Uses Crt,Dos;

Const
        Plus_de_Papier  = 40;  {  56 pour une EPSON LX800 }
        Imp_eteinte     = 200; { 192  "    "    "     "   }
        Imp_ok          = 144; { 144  "    "    "     "   }
        Imp_non_prete   = 8;   {  24  "    "    "     "   }
        Imp_inexistante = 176; { 176  "    "    "     "   }


        tab_etat : array[0..5] of string[22] = ('imprimante ok          ',
                                                'imprimante eteinte     ',
                                                'plus de papier         ',
                                                'imprimante non prète   ',
                                                'imprimante inexistante',
                                                'erreur non repertoriée');

Var
   regs : Registers;
   erreur_imp : byte;


{ procédure effectuant un remise à blanc des registres }

Procedure Efface_Reg;
begin
 with regs do
  begin
   ax:=0;bx:=0;cx:=0;dx:=0;bp:=0;si:=0;di:=0;ds:=0;es:=0;
  end;
end;



Function Imprimante_ok : boolean;
begin
 Efface_reg;
 erreur_imp:=5;
 Regs.AH:=2;        { appel de la fonction n° 2 }
 IntR($17,Regs);    { éxécution de la routine d'interruption  17H }
    case Regs.AH of
        Imp_ok          : erreur_imp:=0;
        Imp_eteinte     : erreur_imp:=1;
        Plus_de_papier  : erreur_imp:=2;
        Imp_non_prete   : erreur_imp:=3;
        Imp_inexistante : erreur_imp:=4;
   end; { du case }
 if Regs.ah=imp_ok then imprimante_ok:=true else imprimante_ok:=false;
end;


begin

  clrscr;
  writeln('attention , ce qui suit n"est valade que pour une NEC P2200');
  writeln;


  if not Imprimante_ok then writeln(tab_etat[erreur_imp])
                       else writeln(tab_etat[erreur_imp]);


end.
```

PASCAL

UN MENU DANS UN FICHIER BAT

Il s'agit de gérer un menu à l'intérieur d'un fichier BAT.

Ceci est réalisé grace à l'utilisation de l'instruction IF ERRORLEVEL N GOTO LABEL, où N est un code de sortie généré par un autre programme.

Ce programme est écrit en Turbo-Pascal v4, le code de sortie est passé en paramètre de la procédure HALT. Les programmes MENU.BAT et MENUA.PAS donnent un exemple de ce qui peut être réalisé.Ils sont adaptables aisément pour d'autres traitements.

PROGRAMME MENUA

La constante RET sert à l'affichage d'une flèche à la queue pliée symbolisant la touche RETOUR CHARIOT. Le tableau de constantes AFF contient les différentes options qui seront affichées par le menu. Celui-ci est adaptable selon les besoins.

La procédure Ecrit(x,y:byte;ch:string;coulfond,coulcar:byte) sert à afficher à la colonne x et à la ligne y la chaîne ch avec les couleurs coulfond pour le fond et coulcar pour le texte. Les couleurs initiales sont restituées grâce à la variable TEXTATTR sauvegardée au début de la procédure dans la variable ATT et restaurée à la fin. La procédure Curs(etat:bool) fait disparaître le curseur si état=0 et le restitue si état=1. Il s'agit d'un appel à l'interruption 16. La procédure Inkey attend la frappe d'un caractère au clavier et retourne dans la variable code le code ASCII de ce caractère, ou le code étendu+256 s'il s'agit d'une touche spéciale: HOME, PgDn etc...

La FUNCTION Affich(y:byte):byte convertit la valeur y de la ligne d'affichage en une valeur pouvant être exploitée en indice par AFF. Sert également pour générer le code de sortie.

La procédure Affichage fait apparaître le menu à l'écran grâce aux constantes du tableau AFF.Une barre en bas donne les touches à utiliser.

La procédure Edite permet le déplacement d'une barre à fond bleu sur les différentes options du menu par utilisation des touches "flèches en haut", "flèches en bas", "Home"(ou "PgUp"), "End(ou "PgDn"). La touche retour valide le choix.

PROGRAMME MENU.BAT

Ce fichier est simple.Il faut faire attention à mettre les IF ERRORLEVEL N par ordre décroissant car le test est inférieur ou égal.

Claude RAYMOND

```
PROGRAM MENUA;

USES
    Crt,
    Dos;

CONST
    ret=#17#196#217;
    aff:array[1..5] of string[19]=('   BASIC            ','   PASCAL           ',
    '   DBASE            ','   MULTIPLAN        ','   RETOUR AU DOS    ');

TYPE
    bool=0..1;

VAR
    sel,ancsel:byte;
    code:integer;
    reg:registers;


PROCEDURE Ecrit(x,y:byte;ch:string;coulfond,coulcar:byte);

Var att:byte;

Begin
    att:=textattr;
    textcolor(coulcar);
    textbackground(coulfond);
    gotoxy(x,y);
    write(ch);
    textattr:=att;
End;
```

```
PROCEDURE Curs(etat:bool);

Var cursouinon:integer;

Begin
    case etat of
      1:cursouinon:=$0607;
      0:cursouinon:=$0B07;
    end;
    reg.ah:=1;
    reg.cx:=cursouinon;
    intr(16,reg);
End;
```

```
ECHO OFF
:DEBUT
MENUA
IF ERRORLEVEL 5 GOTO FIN
IF ERRORLEVEL 4 GOTO ET4
IF ERRORLEVEL 3 GOTO ET3
IF ERRORLEVEL 2 GOTO ET2
IF ERRORLEVEL 1 GOTO ET1
:ET1
CD\BAS
P-BASIC
CD\
GOTO DEBUT
:ET2
CD\PAS                    CD\MP
TURBO                     MP
CD\                       CD\
GOTO DEBUT                GOTO DEBUT
:ET3                      :ET2
CD\DB                     CD\PAS
DBASE                     TURBO
CD\                       CD\
GOTO DEBUT                GOTO DEBUT
:ET4                      :FIN
```

```
PROCEDURE Inkey;

Var ch:char;

Begin
    ch:=readkey;
    if not ((ch=#0) and keypressed)
        then code:=ord(ch)
      else
        begin
          ch:=readkey;
          code:=256+ord(ch);
        end;
End;


FUNCTION Affich(y:byte):byte;

Begin
    affich:=(y-8) div 2+1;
End;


PROCEDURE Affichage;

Var i:byte;

Begin
    ecrit(36,4,'M E N U',0,14);
    ecrit(36,5,'———',0,14);
    for i:=1 to 5 do ecrit(31,(i-1)*2+8,aff[i],0,14);
    ecrit(6,25,'  TOUCHES:  '+chr(24)+' '+chr(25)+'    '+
    ' PgUp PgDn Home End          VALIDATION:  '+ret+'      ',7,1);
End;


PROCEDURE Edite;

Begin
    sel:=8;ancsel:=8;
    curs(0);
    repeat
      ecrit(31,ancsel,aff[affich(ancsel)],0,14);
      ecrit(31,sel,aff[affich(sel)],1,14);
      ancsel:=sel;
      inkey;
      case code of
        336:begin sel:=sel+2;if sel>16 then sel:=8;end;
        328:begin sel:=sel-2;if sel<8 then sel:=16;end;
        327,329:sel:=8;
        335,337:sel:=16;
      end;
    until code=13;
    curs(1);
End;


BEGIN
    clrscr;
    affichage;
    edite;
    clrscr;
    halt(affich(sel));
END.
```

PASCAL

ENVOI DE CARACTERES DE CONTROLE À UNE IMPRIMANTE

Certaines imprimantes possèdent un jeu réduit de boutons qui limite le nombre de fonctions atteintes manuellement. Ce programme vous permet d'envoyer simplement tous les codes que vous désirez.

Le programme créera un fichier de commande à vos spécifications où vous entasserez les codes de contrôles en hexadécimal. On sort de ce programme par appui sur la touche Escape.

Stephane Perroy.

```
program EchoFiles;

Uses
    Dos,Crt;
Var
     unite          : String [3];
     x:char;

procedure BatchFilesDir;

Const
     Maxi           = 50;
Type
    Tableau         = Array [1..Maxi] Of String [12];
Var
   Fichiers         : Tableau;
   Fichier          : SearchRec;
   Nom              : String [12];
   I,J              : Integer;

begin
     for I:=1 to Maxi do fichiers[I]:="";
     FindFirst(unite+'*.BAT',AnyFile,Fichier);
     while DosError=0 do
     begin
          I:=1;
          while (Fichiers[I]<Fichier.Name) And (Fichiers[I]<>"") do
                Inc(I);
          for J:=Maxi-1 DownTo I do
              Fichiers[J+1]:=Fichiers[J];
          Fichiers[I]:=Fichier.Name;
          FindNext(Fichier);
     end;
     ClrScr;
     GotoXY(40,1);
     writeln('FICHIERS D"IMPRESSION BATCH');
     writeln;
     writeln('   Directory :');
     writeln;
     for I:=1 to 10 do
     begin
          for J:=1 to 5 do
          begin
               Nom:=Fichiers[(J-1)*10+I];
               GotoXY((J-1)*15,WhereY);
               write(copy(Nom,1,pos('.',Nom)-1));
          end;
          writeln;
     end;
end;


function CreeFichier : Boolean;
```

```
Var
   Fichier              : File Of Char;
   touche,Sp            : Char;
   Nom                  : String [12];
   Mot                  : String [10];
   I,nb,erreur,Code,n   : Integer;


begin
     CreeFichier:=True;
     repeat
          GotoXY(1,20);
          write('Nom du fichier à créer :     ');
          GotoXY(26,20);
```

```
nb:=0;
nom:="";
repeat
      repeat
            touche:=UpCase(ReadKey);
      until touche in [#13,#27,'A'..'Z','0'..'9'];
if touche in ['A'..'Z','0'..'9'] then begin
   write(touche);
   nb:=nb+1;
   Nom:=Nom+touche;
end
else if touche=#27 then begin
  if nb=0 then begin
  CreeFichier:=False;
  exit;
end;
nb:=0;
nom:="";
```

```
            GotoXY(26,20);
            write('          ');
            GotoXY(26,20);
          end;
        until (touche=#13) or (nb=8);
        Nom:=Nom+'.BAT';
        Assign(Fichier,unite+nom);
        {$I-}
        ReSet(Fichier);
        {$I+}
        touche:='O';
        erreur:=IOResult;
        if erreur<>2 then if erreur<>5 then begin
          GotoXY(1,22);
          write('Le Fichier existe déja. Ecrasement du fichier (O/N) : ? ');
          repeat
            touche:=UpCase(ReadKey);
          until touche in ['O','N'];
          write(touche);
          GotoXY(1,22);
          write('                                                        ');
          end
          else begin
            GotoXY(1,22);
            write('Ouverture du fichier impossible (READ ONLY). Tapez ENTER.');
            readln;
            GotoXY(1,22);
            write('                                                        ');
            touche:='N';
          end;
         {$I-}
         Close(Fichier);
         {$I+}
         erreur:=IOResult;
  until touche='O';
  Assign(fichier,unite+nom);
  ReWrite(fichier);
  Mot:='ECHO OFF'+#13+#10;
  Sp:=' ';
  GotoXY(1,22);
  writeln('ECHO OFF');
  write('ECHO ');
  for I:=1 to 10 do
      write(Fichier,Mot[I]);
  for I:=1 to 4 do
      write(Fichier,Mot[I]);
  write(Fichier,Sp);
  GotoXY(1,25);
  write('Code n°');
  nb:=1;
  repeat
        GotoXY(8,25);
        write(nb,' :    ',#8,#8);
        Code:=0;
        I:=16;
        repeat
```

```
              repeat
                touche:=UpCase(ReadKey);
              until touche in ['0'..'9','A'..'F',#27];
              if touche<>#27 then begin
                If touche in ['A'..'F'] Then n:=10+Ord(touche)-Ord('A')
                Else n:=Ord(touche)-Ord('0');
                Code:=Code+n*I;
                write(touche);
              end;
              I:=I-15;
            until (I<0) Or (touche=#27);
            if touche<>#27 then begin
              if (Code<>8) And (Code<>13) then begin
                GotoXY(5+nb,23);
                write(chr(Code));
              end;
              x:=chr(Code);
              write(Fichier,x);
              nb:=nb+1;
            end;
        until touche=#27;
        Mot:='>PRN'+#13+#10;
        for I:=1 to 4 do
            write(Fichier,Mot[I]);
        Close(Fichier);
  end;


  begin
```

```
    if ParamCount>1 then begin
       writeln('Nombre de paramètres incorrect !');
       writeln('Syntaxe :      printer <drive>');
       halt;
    end;
    if ParamCount=0 then unite:=''
    else begin
      if (ParamStr(1)='A:') or (ParamStr(1)='B:') or (ParamStr(1)='C:') or
      (ParamStr(1)='a:') or (ParamStr(1)='b:') or (ParamStr(1)='c:') then unite:=ParamStr(1)
      else begin
        writeln('Unité de drive incorrect !');
        halt;
      end;
    end;
    repeat
          BatchFilesDir;
    until not CreeFichier;
    ClrScr;
    writeln('Programme Terminé…');
end.
```

BASIC

GESTION BANCAIRE

Même si le sujet n'est pas d'une folle excitation, il reste parfois utile de gérer ses flux financiers.

Vous trouverez de même une illustration de l'emploi des touches de fonction ainsi que le traitement des erreurs.

Roland Lopez

D'autant que le programme peut s'adapter à d'autres besoins et remanier à loisir car il est écrit en Basic sans finesse incompréhensible.

Le programme comporte cinq fonctions.

La création du fichier que l'on effectue une fois pour toutes. La mise à jour où l'on comptabilise les flux entrées et sorties encompte courant ou en dépôt. La suppression d'un mouvement par sonnuméro. La visualisation du fichier par mois et type de compte.
C'est là qu'apparaît aussi le numéro de mouvement. L'impressiondu fichier. Le fichier contenant les mouvements est sur le lecteur par défaut.
Attention le programme comporte une instruction qui appelle la fonction ATTRIB du DOS par un Shell. Il est bon de recopier ce fichier sur votre disquette. Ce fichier permet de modifier les attributs des fichiers (Read-only, Hidden, etc). C'est un bonexemple de son utilisation.

```
100 WIDTH 80:CLEAR:KEY OFF
110 ON ERROR GOTO 2780
120 '
130 ' ----- Code -----
140 '
150 ESSAI=0
160 COLOR 3,0
170 CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,24:PRINT
"G E S T I O N   B A N C A I R E":COLOR 3,0
180 LOCATE 10,30:PRINT "CODE D'ACCES :":COLOR 0,7
190 LOCATE 10,45:LINE INPUT CODE$:COLOR 3,0
200 IF CODE$="" THEN 295
210 IF CODE$="KILROY" OR CODE$="kilroy" THEN 360
220 COLOR 31,9:ESSAI=ESSAI+1
230 LOCATE 10,45:PRINT "ACCES REFUSE":PRINT CHR$(7):COLOR 3,0
240 FOR T=1 TO 2500:NEXT T
250 IF ESSAI=3 THEN 260 ELSE 170
260 CLS:COLOR 31,9
270 LOCATE 10,27:PRINT ">>   IDENTIFICATION INCORRECTE   <<"
280 LOCATE 12,32:PRINT ">>   CLAVIER  BLOQUE   <<"
290 GOTO 290
295 LOCATE 19,7:PRINT "┌──────────────
```

UN GENERATEUR DE TABLEAUX

Si vous demandez à des dactylos ou secrétaires quelle est leur hantise, la plupart vous répondrons «les tableaux !».

En effet les largeurs de colonnes à retoucher fréquemment, le tracé des traits horizontaux et verticaux, les valeurs numériques à "aligner", les tableaux qui deviennent trop larges, constituent une rude épreuve pour les nerfs. Or paradoxalement il n'existe pas de logiciels pour cela ; si ce n'est les dernières versions de Sprint, Word et WordPerfect qui facilitent ce problème.

PRESENTATION DU PROGRAMME "TABLIC".

Le rôle de ce logiciel est d'imprimer des tableaux dont les valeurs ont été entrées au clavier.
A l'inverse des logiciels du commerce que l'on doit apprendre, ici c'est le programmeur qui a dû apprendre les voeux des utilisateurs. Un exemple, on part toujours d'un original manuscrit et souvent plein de ratures et surcharges : De ce fait on peut oublier une ligne, ou bien l'écrire deux fois de suite, mais il est impossible de se tromper sur le nombre de colonnes ; donc pas de correction de ce nombre. En revanche beaucoup d'exigences et de souplesse pour ce qui concerne la présentation finale de l'impression. Bref un outil de bureautique et non un de ces jouets à mille facettes.
Ce n'est pas un tableur. Les "cases" du tableau sont saisies une à une, par ligne et par colonne. C'est le programme qui calcule les largeurs de colonnes, qui trace les traits, qui assure les diverses "justifications" et la taille des caractères. En somme «entrez les contenus des cases, je me charge du reste».
- Le titre (facultatif) est auto-centré au dessus du tableau. - Les légendes de colonnes sont centrées. - Les contenus des colonnes "texte" sont alignés à gauche. - Les contenus des colonnes "nombres" sont justifiés à droite et sur la position du point décimal. - Les traits (caractères graphiques) peuvent être simples, doubles ou remplacés par des blancs. - Insertion de lignes de séparations : ligne vide ou trait horizontal. - Centrage facultatif de l'ensemble du tableau. - Toute correction provoque le recalcul des largeurs de chacune des colonnes. - Choix de la largeur du papier. Exemple «normale en caractères condensés ou à l'Italienne en caractères pica ». - Possibilité d'imprimer une version du tableau où colonnes et lignes sont numérotées ; pour mieux identifier les cases à corriger. - Sauvegarde uniquement sur disquette, ce pour la portabilité sur un poste équipé d'une imprimante à chariot large ou d'une laser configurée en mode "Landscape". - La sauvegarde fabrique deux fichiers : un *.VAL (valeurs saisies) lisible seulement par TABLIC, et un *.TAB qui est la représentation ASCII du tableau imprimé. Ce dernier peut donc être visionné ou imprimé à partir du DOS, exemple TYPE A:TOTO.TAB>PRN - Ce fichier .TAB peut donc être appelé pour insertion à partir d'un traitement de texte. Exemple avec WORD : "Lit écrit" "Fusionne" a:toto.tab - Possibilité de sauvegarder un *.TAB sans les traits graphiques; afin qu'un informaticien puisse récupérer ces colonnes de nombres dans un logiciel graphique ou un tableur. - Le programme est écrit en GWBASIC (15 kilos octets) mais compilable ; c'est même vivement recommandé afin de gagner en mémoire disponible.
Et maintenant LA lacune : Si le bandeau des légendes de colonnes peut comporter plusieurs lignes, il ne peut y avoir de légendes arborescentes ; c'est à dire de colonnes larges se divisant plus bas en plusieurs plus petites. Il est plus facile et plus clair de rajouter cé «chapô» sous traitement de texte.
Nous pouvons dès lors passer au mode d'emploi :

LE MENU PRINCIPAL.

Six options :
N = nouveau tableau : définitions et saisie des valeurs. S = correction de valeurs entrées ou ajouts, suppressions de lignes. I = impression. S = sauvegarde sur une disquette formatée dans A: R = recharger un ancien fichier tableau depuis A: Q = quitter le programme.

On demande d'abord le titre à placer au dessus du tableau ; c'est facultatif.
Ensuite, nombre de colonnes : Attention ! Cette valeur ne sera plus modifiable par la suite.
Nombre de lignes de légendes ; minimum une... Ces lignes ont un statut à part : Leurs contenus sont centrés automatiquement dans les "cellules" et un trait continu sera généré sous elles. Très souvent la deuxième ligne de légendes est prévue pour y inscrire les unités (F HT, g/cm˜, etc...)

Questionnaire sur les légendes : colonne par colonne on demande leur NATURE. Si c'est du texte répondre T (ou t), pour des nombres mettez seulement le nombre de décimales souhaitées, de 0 à 6. Ces options sont rappelées à l'écran. Elles sont modifiables en cas d'erreur. Puis vous devrez entrer les légendes proprement dites, par colonne puis par ligne. Pour une "case" vide tapez seulement ENTER.

Saisie des valeurs : Afin d'éviter les erreurs l'écran affiche le N° de ligne, le N° de colonne et la ou les légendes de cette colonne. QUATRE OPTIONS sont accessibles, seulement en colonne N°1:
- T (ou t) : cette ligne sera un trait horizontal de séparation. - = cette ligne sera vide (une séparation). - E (ou e) pour "erreur": On peut retaper alors la ligne précédente. - Q (ou q) pour quitter la saisie, ne serait-ce que pour une sauvegarde avant de poursuivre la saisie.
NOTA : Si l'un de ces caractères est suivi ou précédé d'un espace il ne sera pas considéré comme option.
Sitôt la fin de la saisie TABLIC commence ses calculs : Pour chaque colonne il recherche l'élément le plus long pour établir les largeurs réelles. (Il laisse un espace de chaque côté pour ne pas "coller" aux traits verticaux). Puis il calcule les tabulations et la largeur totale "hors tout". Si celle-ci dépasse 132 caractères vous en serez avisé, avec le choix de corriger ou de maintenir.
Le corrigé vous permet de modifier le titre, les légendes (et leur « nature ») et les contenus des "cases" du tableau. Un menu propose alors la suppression ou l'insertion de ligne, la "reprise" de toute une ligne, ou d'une case déterminée.
Vous devez entrer les N° de colonne et de

ligne. L'écran affiche alors la légende de la colonne, le contenu de la colonne N°1 de cette ligne et le contenu actuel de la "cellule" (ou "case") concernée. Tapez la correction, ou un "ENTER à vide" qui laissera l'ancienne valeur. Pour effacer le contenu d'une case TAPEZ ESPACE et ENTER.
En fin de correction (toujours Q ou q pour quitter) le logiciel reprend tous ses calculs à zéro. (c'est très rapide...).
Petit rappel, pour ajouter des lignes (suite de saisie) il faut repasser par menu, puis option corriger, laquelle vous propose alors Correction ou Suite de saisie.

LES COLONNES de VALEURS NUMERIQUES.

Si à la rubrique Légendes une colonne a été définie par exemple avec «nature = 2», tous les nombres y seront imprimés avec 2 décimales. Ainsi 0 23 34.5 45.678 donneront respectivement 0.00 23.00 34.50 45.68. Quelques remarques : -La valeur est arrondie sur le dernier chiffre (45.678-->45.68) mais la valeur tapée sera néanmoins sauvegardée. -Les nombres décimaux utilisent le POINT.(virgule ignorée) -Bien sûr les nombres sont présentés "alignés". (à droite) -Dans une colonne de nature numérique on peut taper du texte ; mais il sera "justifié à gauche". Un "ENTER à vide" laissera la cellule vide.

L'EDITION SUR IMPRIMANTE.

Selon la largeur du tableau, TABLIC commande les tailles PICA, ELITE ou CONDENSE, mais pas avec les imprimantes LASER dépourvues d'émulation "EPSON FX". Si la largeur dépasse 78 caractères l'écran vous propose l'impression à l'Italienne (papier dans le sens longueur).
Les options "caractères renforcés" ou "qualité courrier" (NLQ, ROMAN) pourront être préalablement fixées par les touches de l'imprimante. Surtout pas d'espacement proportionnel ! (PS)
Au lancement de l'impression vous disposez de plusieurs options : "I" impression normale sans centrage ; "L" idem mais avec numérotation des lignes et colonnes (pour modifier) ; "C" pour le centrage horizontal du tableau sur le papier. Ensuite le choix des traits, simples, doubles ou sans.

LA SAUVEGARDE du TABLEAU.

Elle est facultative, mais conseillée. Et même rappelée si vous quittez le logiciel sans l'avoir faite. Ne pas donner de nom d'extension, à cause des .VAL et .TAB. Même si vous avez imprimé avec centrage, le fichier .TAB n'aura pas de blancs à sa gauche, car ils seraient très gênants en cas de fusion dans un traitement de texte.

LE PROGRAMME BASIC.

Nous ne ferons pas de description détaillée, car les REM sont nombreux, mais des explications sur certains détails. - 4240 : PU$ est un format pour imprimer en PRINT USING. - 7100 : afin que la sauvegarde et l'impression utilisent le même sous-programme en 13000, pas de LPRINT mais déclaration de LPT1: en canal #2. - 10100 à 10250 : ce passage (facultatif) propose de copier le .TAB dans votre répertoire de traitement de texte sur disque dur avec changement du nom d'extension. L'exemple concerne ici WORD pour un répertoire C:\WORD\TEXTE avec extension .DOC. Vous adapterez donc à votre besoin. - 39000 à 64000 : il s'agit de divers modules utilitaires qui ont été rechargés par MERGE. Sauvegardez-les aussi séparément pour les utiliser dans vos programmes personnels.

Vous remarquerez la profusion de sécurités diverses, car un plantage dû à une fausse manoeuvre de l'opérateur serait une catastrophe : Quand un programme compilé (.EXE) plante il y a retour au DOS, donc avec perte de toutes les données...

TRES IMPORTANT : En cas d'imprimante lente ou dépourvue de buffer il sera très prudent de neutraliser le "timeout" de la sortie du PC, d'où ce TAB.BAT de lancement.

MODE LPT1:80,6,P TABLIC.EXE (ou GWBASIC TABLIC.BAS) MODE LPT1:80,6

Michel Archambault

```
10 ' TABLIC - Edition automatique de tableaux - Michel Archambault - 21/02/89
20 ' directement compilable par TURBO BASIC de BORLAND
30 DEFINT A-Z:KEY OFF:OPTION BASE 0
40 ON ERROR GOTO 64000
50 FOR I=11 TO 14:KEY(I) ON:NEXT:'neutralise touches curseur
100 ' DIM en 2070 ou 12125 : NC$=Natures des Colonnes ; LC=Largeurs des Colonnes
 ; PU$=formats PRINT USING ; TV=Tabulations verticales
200 ' codes des caractères graphiques pour traits
210 DATA 218,191,192,217,179,196,195,180,194,193,197
220 DATA 201,187,200,188,186,205,204,185,203,202,206
230 DATA 32,32,32,32,32,32,32,32,32,32,32
240 FOR N=1 TO 3:READ A(N),B(N),C(N),D(N),E(N),F(N),G(N),H(N),I(N),J(N),K(N):NEX
T
1000 'MENU
1010 DATA "Nouveau tableau","Corrigé, Suite de saisie",Impression,"Sauvegarde su
r A:","Recharge un tableau en A:",Quitter
1015 CLS:YC=1:LC=43:NLI=5:GOSUB 39700
1020 LOCATE 4,25:PRINT"Edition automatique de Tableaux"
1030 LOCATE 6,19:PRINT"(Michel Archambault - version février 89)"
1040 IF NC>0 THEN LOCATE 20,2:PRINT FI$:LOCATE 22,1:PRINT NL;"lignes"
1090 TIM$="T  A  B  L  I  C":NOPT=6:RESTORE 1010:GOSUB 49000
```

```
1100 ON K GOTO 2000,8000,7000,9000,12000,1500
1500 'FIN
1510 IF FCALC=0 THEN 1600
1520 CLS:LOCATE 12,30:PRINT"Voulez-vous sauvegarder ?":TEX$="ON":GOSUB 50000:ON
K GOTO 9000,1600
1600 ON ERROR GOTO 0:CLS:END
2000 'SAISIE
2010 IF NC>0 THEN 2700
2020 CLS:YE=2:E$="Création d'un nouveau Tableau.":GOSUB 39200
2025 LOCATE 7,20:PRINT"(facultatif, ou Q pour quitter)"
2030 LOCATE 5,2:LINE INPUT"TITRE du TABLEAU : ",TIT$:IF TIT$="" THEN 2050
2040 IF INSTR("Qq",TIT$)>0 THEN 1000
2050 LOCATE 9,30:INPUT"Nombre de Colonnes : ",NC$
2060 NC=VAL(NC$):IF NC<2 THEN 2050
2070 DIM NC$(NC),LC(NC),PU$(NC),TV(NC+1),V$(300,NC)
2080 LOCATE 12,8:LINE INPUT"Combien de Lignes pour les Légendes de Colonnes ? (1
,2,3...) : ",NLG$
2090 NLG=VAL(NLG$):IF NLG<1 THEN BEEP:GOTO 2080
2100 L=0:FOR C=1 TO NC
2110 CLS:LOCATE 1,3:PRINT"NATURE : si Texte répondre T ; si numérique tapez le n
ombre de décimales.":L=0
2120 E$="Colonne N°"+STR$(C):YE=5:GOSUB 39000
2140 GOSUB 21000
2200 ON K GOTO 2210,2120
2210 NEXT:V$(NLG+1,1)="T":NL=NLG+1:GOTO 3000
2700 'Sécurité effacement
2710 CLS:BEEP:LOCATE 10,2:PRINT"La Création d'un nouveau Tableau va effacer de l
a Mémoire le Tableau actuel !"
2720 LOCATE 14,30:PRINT"Confirmez-vous cet Ordre ?":TEX$="ON":GOSUB 50000
2730 IF K=1 THEN RUN
2740 GOTO 1000
3000 ' ENTREES DES VALEURS
3005 GOSUB 20000
3010 CLS:YE=2:E$="ENTREES des VALEURS du TABLEAU":GOSUB 39000
3020 LOCATE 5,1:PRINT"Nous allons procèder ligne par ligne (les légendes sont dé
jà faites) , puis     colonne par colonne."
3030 LOCATE 9,2:PRINT"Seront affichés : N° Ligne, N° Colonne et la ou les légend
e(s) de la colonne."
3040 LOCATE 11,13:PRINT"En COLONNE N° 1 (uniquement) on dispose de QUATRE OPTION
S :":PRINT
3050 PRINT"   T fera un trait horizontal de séparation.":PRINT
3060 PRINT"   = fera une ligne vide, pour séparation.":PRINT
3070 PRINT"   E pour erreur : on reprend alors la ligne précèdente.":PRINT
3080 PRINT"   Q pour quitter la saisie."
3090 LOCATE 22,36:INPUT"Tapez ENTER ",R$:L=NL
3100 L=L+1:FQ=0:CLS
3110 GOSUB 22000
3200 IF FQ=0 THEN 3300
3210 DATA "Fin de saisie",Corriger,"Suite de la saisie"
3220 L=L-1:I=I-1:NL=L:RESTORE 3210:CLS:NOPT=3:TIM$="":GOSUB 49000
3230 ON K GOTO 3900,8000,3100
3300 GOTO 3100
3900 NL=L
4000 ' CALCUL DES LARGEURS DE COLONNES
4010 CLS:LOCATE 12,35:PRINT"Patience...":FCALC=1:FOR C=1 TO NC:LC(C)=1:NEXT
4020 FOR C=1 TO NC:FOR L=1 TO NLG:S=LEN(V$(L,C)):IF S>LC(C) THEN LC(C)=S
4030 NEXT:NEXT:FOR C=1 TO NC:IF NC$(C)<>"T" THEN 4200
4040 FOR L=NLG+2 TO NL:S=LEN(V$(L,C)):IF S>LC(C) THEN LC(C)=S
4050 NEXT L:GOTO 4300
4200 ND=VAL(NC$(C)):FOR L=NLG+2 TO NL
4210 S=LEN(STR$(INT(VAL(V$(L,C)))))+VAL(NC$(C))-(ND>0)-1:IF VAL(V$(L,C))<0 THEN
S=S+1
4220 IF S>LC(C) THEN LC(C)=S
```

```
4230 NEXT L
4240 PU$(C)=STRING$(LC(C),"#")
4250 IF ND>0 THEN MID$(PU$(C),LC(C)-ND,1)="."
4300 NEXT C
4500 'Largeur totale LT
4510 LT=0:FOR C=1 TO NC:LT=LT+LC(C):NEXT:LT=LT+NC*3+1
4520 IF LT<132 THEN 4600
4530 CLS:BEEP:LOCATE 10,2:PRINT"ATTENTION ! Largeur totale =";LT;"caractères. (m
axi=132 en A4 et 195 en A3)"
4540 LOCATE 13,24:PRINT"Voulez-vous CORRIGER ou MAINTENIR ?":TEX$="CM":GOSUB 500
00
4550 ON K GOTO 8000,4590
4590 CLS:LOCATE 12,35:PRINT"Patience..."
4600 TD=1
5000 'TAB TRAITS VERTICAUX
5010 TV(1)=TD
5020 FOR C=1 TO NC:TV(C+1)=TV(C)+LC(C)+3:NEXT
5030 GOTO 1000
7000 ' IMPRESSION
7010 IF NC=0 OR NL=0 THEN BEEP:GOTO 1000
7020 FQM=0:GOSUB 15000:IF FQM=1 THEN 1000
7030 GOSUB 27000:IF FQI=1 THEN 7120
7040 GOSUB 16000
7050 CLS:LOCATE 12,32:PRINT"IMPRESSION en COURS":WIDTH "LPT1:",255:LPRINT CI$
7100 OPEN "LPT1:" FOR OUTPUT AS #2:Z=2
7110 GOSUB 13000:CLOSE#2
7120 GOTO 1000
8000 'CORRIGE
8010 FCOR=0:IF NC=0 OR NL=0 THEN BEEP:GOTO 1000
8015 DATA Corriger,"Suite de la Saisie",Quitter
8020 CLS:RESTORE 8015:TIM$="":NOPT=3:GOSUB 49000:ON K GOTO 8030,3000,1000
8030 CLS:LOCATE 10,2:PRINT"Titre actuel : ";TIT$
8040 LOCATE 14,2:LINE INPUT"Corrigez (ENTER = inchangé) : ",R$
8050 IF R$<>"" THEN TIT$=R$:FCOR=1
8060 DATA "Insérer une ligne","Supprimer une ligne","Modifier le contenu d'une c
ase","Ressaisir toute une ligne",Quitter
8100 CLS:E$="CHOIX du TYPE de CORRECTION":YE=5:GOSUB 39000:LOCATE 1,1:PRINT NL;"
lignes au total":RESTORE 8060:NOPT=5:TIM$="":GOSUB 49000
8110 ON K GOTO 8200,8300,8500,8400,8120
8120 IF FCOR=1 THEN 4000 ELSE GOTO 1000
8200 'insert
8210 CLS:LOCATE 10,19:LINE INPUT"Insertion APRES quel N° de Ligne ? (ou Q) ",M$
8220 IF M$="Q" OR M$="Q" THEN 8100 ELSE M=VAL(M$)
8230 IF M<1 OR M>NL THEN GOSUB 25000:GOTO 8200
8240 IF M<NLG+1 THEN BEEP:LOCATE 16,13:PRINT"ATTENTION ! ce sera une ligne légen
de. Confirmez-vous ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 8200 ELSE NLG=NLG+1
8250 LOCATE 12,36:PRINT"Patience...":GOSUB 23000:FCOR=1
8260 CLS:L=M+1:LOCATE 1,24:PRINT"Saisie de cette nouvelle LIGNE N°";L:PRINT:GOSU
B 22020
8270 GOTO 8100
8300 'supprim
8310 CLS:LOCATE 8,19:LINE INPUT"Suppression de quel N° de Ligne ? (ou Q) ",M$
8320 IF M$="Q" OR M$="Q" THEN 8100 ELSE M=VAL(M$)
8330 IF M<1 OR M>NL OR M=NLG+1 THEN GOSUB 25000:GOTO 8300
8340 IF M<NLG+1 THEN BEEP:LOCATE 12,14:PRINT"ATTENTION ! c'est une ligne légende
. Confirmez-vous ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 8300 ELSE NLG=NLG-1
8350 LOCATE 14,24:PRINT"Contenu de la colonne N° 1 : ";V$(M,1);"  OK ?":TEX$="ON
":GOSUB 50000:IF K=2 THEN 8300
8360 LOCATE 17,36:PRINT"Patience...":GOSUB 24000:FCOR=1:GOTO 8100
8400 'ressaisie ligne
8410 CLS:LOCATE 10,24:LINE INPUT"N° de la Ligne à ressaisir ? (ou Q) ",M$
8420 IF M$="Q" OR M$="Q" THEN 8100 ELSE M=VAL(M$)
8430 IF M<1 OR M>NL OR M=NLG+1 THEN GOSUB 25000:GOTO 8400
```

```
8440 LOCATE 13,2:PRINT"Contenu de la case LIGNE";M;"colonne 1 : ";V$(M,1):LOCATE
 15,38:PRINT"OK ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 8400
8450 IF M<NLG+1 THEN BEEP:LOCATE 16,14:PRINT"ATTENTION ! c'est une ligne légende
. Confirmez-vous ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 8400
8460 CLS:L=M:GOSUB 22000:FCOR=1:GOTO 8100
8500 'case
8510 CLS:YE=2:E$="Modification du contenu d'une case":GOSUB 39000
8520 LOCATE 5,26:LINE INPUT"Quel N° de Colonne ? (ou Q) ",C$
8530 IF C$="Q" OR C$="q" THEN 8100 ELSE C=VAL(C$)
8540 IF C<1 OR C>NC THEN GOSUB 25000:GOTO 8500
8550 LOCATE 7,10:PRINT"Légende = ";:FOR I=1 TO NLG:PRINT V$(I,C);"/";:NEXT:PRINT
"   OK ?":TEX$="ON":GOSUB 50000:IF K=2 THEN 8500
8560 LOCATE 10,20:LINE INPUT"Quel N° de Ligne ? ",M$:M=VAL(M$)
8570 IF M<1 OR M=NLG+1 OR M>NL OR V$(M,1)="T" THEN GOSUB 25000:LOCATE 18,29:PRIN
T SPACE$(28):GOTO 8560
8580 IF M<NLG+1 THEN BEEP:LOCATE 12,14:PRINT"ATTENTION ! c'est une ligne légende
. Confirmez-vous ?":TEX$="ON":GOSUB 50000:IF K=2 THEN CLS:GOTO 8560
8590 IF C>1 THEN LOCATE 14,24:PRINT"Contenu de la colonne N° 1 : ";V$(M,1);"  OK
 ?":TEX$="ON":GOSUB 51000:IF K=2 THEN 8500
8600 '
8610 LOCATE 16,15:PRINT"Contenu de la case demandée : ";V$(M,C)
8620 LOCATE 18,7:LINE INPUT"Nouveau contenu (ENTER si inchangé) : ",R$
8630 IF R$<>"" THEN V$(M,C)=R$:FCOR=1
8635 IF M=1 THEN GOSUB 26000:GOTO 8100
8640 IF C=1 AND R$="t" THEN V$(M,C)="T"
8650 GOTO 8100
9000 ' SAUVEGARDE
9005 IF NC=0 OR NL=0 THEN BEEP:GOTO 1000 ELSE GOSUB 11000
9010 CLS:LOCATE 12,16:INPUT"Préparez une DISQUETTE FORMATEE en Drive A:. ENTER "
,R$
9020 OPEN FICT$ FOR OUTPUT AS #1:Z=1:FL=0:' sauvegarde .TAB
9030 GOSUB 13000:CLOSE #1
10000 ' Sauvegarde .VAL
10010 OPEN FICV$ FOR OUTPUT AS #1
10020 PRINT#1,TIT$:PRINT#1,NC:PRINT#1,NL:PRINT#1,NLG
10030 FOR C=1 TO NC:PRINT#1,NC$(C):NEXT
10040 FOR C=1 TO NC:FOR L=1 TO NL:PRINT#1,V$(L,C):NEXT:NEXT
10050 CLOSE#1:FCALC=0
10100 ' *.TAB vers WORD en *.DOC
10110 CLS:LOCATE 12,7:PRINT"Voulez-vous copier ce fichier dans votre Répertoire
de textes WORD ?":TEX$="ON":GOSUB 50000
10120 ON K GOTO 10200,1000
10200 LOCATE 15,2:LINE INPUT"Répertoire C:\WORD\TEXTE  Corrigez ou ENTER ",R$
10210 PAT$="C:\WORD\TEXTE":IF R$<>"" THEN PAT$=R$
10220 FIW$=PAT$+"\"+FI$+".DOC"
10230 SHELL"COPY "+FICT$+" "+ FIW$
10240 LOCATE 17,12:PRINT"Lit écrit, Fusionne, puis ";FI$;".DOC";:INPUT" . ENTER
",R$
10250 GOTO 1000
11000 ' NOM FICHIER SAUVEGARDE
11010 IF FI$="" THEN 11100
11020 CLS:LOCATE 10,20:PRINT"Sauvegarde du Tableau sous le Nom de ";FI$;" ?":TEX
$="ON":GOSUB 50000
11030 ON K GOTO 11500,11100
11100 CLS:YE=5:E$="SAUVEGARDE de ce FICHIER":GOSUB 39200
11110 LOCATE 8,3:PRINT"Tapez un Nom de HUIT caractères MAXI, sans EXTENSION ni n
om de DRIVE (A:)"
11120 LOCATE 10,1:PRINT"Deux fichiers seront créés en A: .VAL pour les valeurs e
t .TAB pour le Tableau."
11130 LOCATE 13,42:PRINT SPACE$(8);">       ":LOCATE 13,30:LINE INPUT"NOM (ou Q):
 ",F$
11135 IF F$="Q" OR F$="q" THEN 1000
11140 IF LEN(F$)>8 OR F$="" THEN BEEP:GOTO 11130
```

```
11150 IF INSTR(F$," ")+INSTR(F$,".")+INSTR(F$,":")>0 THEN BEEP:GOTO 11130
11500 FI$=F$:FICT$="A:"+FI$+".TAB":FICV$="A:"+FI$+".VAL"
11510 RETURN
12000 ' CHARGE
12005 IF NC>0 THEN 2700
12010 CLS:YE=6:E$="CHARGEMENT d'un FICHIER en Drive A: (Q=quitter, L=liste des f
ichiers)":GOSUB 39200
12020 LOCATE 14,3:PRINT"Tapez un Nom de HUIT caractères MAXI, sans EXTENSION ni
nom de DRIVE (A:)"
12030 LOCATE 12,39:PRINT SPACE$(8);">":LOCATE 12,33:LINE INPUT"NOM : ",UPP$:GOSU
B 51000:F$=UPP$
12035 IF F$="Q" THEN 1000 ELSE IF F$="L" THEN GOSUB 12500:GOTO 12035
12040 IF LEN(F$)>8 OR F$="" THEN BEEP:GOTO 12030
12050 IF INSTR(F$," ")+INSTR(F$,".")+INSTR(F$,":")>0 THEN BEEP:GOTO 12030
12060 FI$=F$:FICV$="A:"+FI$+".VAL"
12100 CLS:LOCATE 12,35:PRINT"Patience..."
12110 OPEN FICV$ FOR INPUT AS #1
12120 LINE INPUT#1,TIT$:INPUT#1,NC:INPUT#1,NL:INPUT#1,NLG
12125 DIM NC$(NC),LC(NC),PU$(NC),TV(NC+1),V$(200,NC)
12130 FOR C=1 TO NC:INPUT#1,NC$(C):NEXT
12140 FOR C=1 TO NC:FOR L=1 TO NL:LINE INPUT#1,V$(L,C):NEXT:NEXT
12150 CLOSE#1
12160 GOTO 4020
12500 ' liste A:
12510 CLS:PRINT TAB(15);"Noms de Fichiers Tableaux présents sur Disquette en A:"
:PRINT
12520 FILES"A:*.VAL"
12530 PRINT TAB(18);"";:LINE INPUT"NOM (sans extension ni nom de Drive) ",UPP$:G
OSUB 51000:F$=UPP$
12540 RETURN
13000 ' IMPRESSION/SAUVEGARDE du Tableau
13010 PRINT#Z,TAB((LT-LEN(TIT$))/2+TD-1);TIT$
13020 IF FL=1 THEN FOR C=1 TO NC-1:PRINT#Z,TAB(TV(C)-1+(TV(C+1)-TV(C))/2);C;:NEX
T:PRINT#Z,TAB(TV(NC)-1+(LT-TV(NC))/2);NC
13030 PRINT#Z,TAB(TD);CHR$(A(D));:FOR C=1 TO NC-1
13040 PRINT#Z,STRING$(LC(C)+2,F(D));CHR$(I(D));:NEXT
13050 PRINT#Z,STRING$(LC(NC)+2,F(D));CHR$(B(D))
13060 L$="":FOR L=1 TO NLG:IF FL=1 THEN L$=STR$(L):'légendes centrées
13070 FOR C=1 TO NC
13080 PRINT#Z,TAB(TV(C)+TD-1);CHR$(E(D));SPACE$((LC(C)-LEN(V$(L,C)))/2+1);V$(L,C
);:NEXT
13090 PRINT#Z,TAB(TV(NC+1)+TD-1);CHR$(E(D));L$:NEXT
13100 ' ligne valeurs
13110 FOR L=NLG+1 TO NL:IF FL=1 THEN L$=STR$(L)
13120 IF V$(L,1)="T" THEN GOSUB 13300:GOTO 13190
13130 IF V$(L,1)="=" THEN GOSUB 13400:GOTO 13190
13140 FOR C=1 TO NC
13150 PRINT#Z,TAB(TV(C)+TD-1);CHR$(E(D));" ";
13160 IF NC$(C)<>"T" AND (VAL(V$(L,C))<>0 OR V$(L,C)="0") THEN PRINT#Z,USING PU$
(C);VAL(V$(L,C));:GOTO 13180
13170 PRINT#Z,V$(L,C);
13180 NEXT C:PRINT#Z,TAB(TV(NC+1)+TD-1);CHR$(E(D));L$
13190 NEXT L
13200 PRINT#Z,TAB(TD);CHR$(C(D));:FOR C=1 TO NC-1
13210 PRINT#Z,STRING$(LC(C)+2,F(D));CHR$(J(D));:NEXT
13220 PRINT#Z,STRING$(LC(NC)+2,F(D));CHR$(D(D))
13230 IF Z=2 THEN PRINT#Z,CHR$(27);"@"
13240 TD=1:RETURN
13300 'trait horizontal
13310 PRINT#Z,TAB(TD);CHR$(G(D));:FOR C=1 TO NC-1
13320 PRINT#Z,STRING$(LC(C)+2,F(D));CHR$(K(D));:NEXT
13330 PRINT#Z,STRING$(LC(NC)+2,F(D));CHR$(H(D));L$
13340 RETURN
```

```
13400 'ligne vide
13410 FOR C=1 TO NC:PRINT#Z,TAB(TV(C)+TD-1);CHR$(E(D));:NEXT
13420 PRINT#Z,TAB(TV(NC+1)+TD-1);CHR$(E(D));L$
13430 RETURN
15000 'TAILLES CARACTERES en Orientations normale & à l'italienne
15010 'indice 1=normal ; indice 2=italienne ; CI$()=CODE EPSON ; CPL()=carac. pa
r ligne ; CAR$()=taille en clair
15020 CI$(1)="":CI$(2)="":CAR$(1)="pica":CAR$(2)="pica":CPL(1)=80:CPL(2)=115
15030 IF LT<=78 THEN K=1:GOTO 15450
15040 IF LT<=95 THEN CI$(1)=CHR$(27)+"M":CPL(1)=96:CAR$(1)="élite":GOTO 15400
15050 IF LT<=115 THEN CI$(1)=CHR$(15):CPL(1)=137:CAR$(1)="condensé":GOTO 15400
15060 IF LT<=137 THEN CI$(2)=CHR$(27)+"M":CPL(2)=137:CAR$(2)="élite"
15070 IF LT<=195 THEN CI$(2)=CHR$(15):CPL(2)=193:CAR$(2)="condensé"
15300 'CHOIX FORMAT
15310 CLS:BEEP
15320 LOCATE 8,32:PRINT"A T T E N T I O N  !"
15330 LOCATE 12,2:PRINT"Ce Tableau ne peut être imprimé qu'à l'ITALIENNE ! (Larg
eur";LT;"caractères)"
15340 LOCATE 16,26:PRINT"OK ou Modifications du tableau ?":TEX$="OM":GOSUB 50000
:IF K=2 THEN FQM=1:GOTO 15900
15350 K=2:GOTO 15450
15400 CLS:BEEP:LOCATE 6,22:PRINT"Quelle LARGEUR PAPIER choisissez-vous ?"
15410 LOCATE 9,21:PRINT"(le tableau fait";LT;"caractères de large)"
15420 LOCATE 12,20:PRINT"N - NORMALE en caractères ";CAR$(1)
15430 LOCATE 14,20:PRINT"I - à l'ITALIENNE en caractères ";CAR$(2)
15440 TEX$="NI":GOSUB 50000
15450 CI$=CI$(K):CPL=CPL(K)
15900 RETURN
16000 'EPAISS.TRAIT
16010 DATA Simples,Doubles,"Pas de traits"
16020 RESTORE 16010:CLS:NOPT=3:TIM$="Epaisseur des Traits":GOSUB 49000:D=K:CLS
16030 RETURN
20000 ' NOTA LIGNES
20010 CLS:YE=2:E$="NOTRE NUMEROTATION DES LIGNES":GOSUB 39200
20020 LOCATE 5,1:PRINT"Vous avez prévu";NLG;"lignes de légendes de colonnes."
20030 LOCATE 7,1:PRINT"TABLIC se charge de tracer un trait horizontal de séparat
ion sur la ligne N°";NLG+1
20040 LOCATE 9,1:PRINT"Votre première ligne de données sera donc la ligne N°";NL
G+2
20050 LOCATE 11,1:PRINT"Si plus loin vous intercalez un autre trait ou une ligne
 blanche de séparation  celle-ci comptera pour une ligne."
20060 LOCATE 15,1:PRINT"En somme la numérotation des lignes commence à la premiè
re ligne de légende."
20070 LOCATE 17,1:PRINT"En fin de tableau TABLIC se chargera de tracer le trait
final."
20080 LOCATE 19,1:PRINT"L'option L à l'impression fournit une numérotation des l
ignes."
20100 LOCATE 22,36:INPUT"TAPEZ ENTER ",R$
20110 RETURN
21000 'SAISIE ligne LEGENDE
21005 FOR L=1 TO NLG
21010 LOCATE 6+L*2,3:PRINT"Légende ligne";L;:LINE INPUT": ",V$(L,C):NEXT
21020 LOCATE 8+NLG*2,29:LINE INPUT"NATURE (T ou 0....6) : ",N$
21030 IF LEN(N$)<>1 OR INSTR("Tt0123456",N$)=0 THEN BEEP:GOTO 21020
21040 NC$(C)=N$:IF N$="t" THEN NC$(C)="T"
21050 LOCATE 20,35:PRINT"Est-ce OK ?":TEX$="ON":GOSUB 50000
21060 RETURN
22000 'SAISIE LIGNE
22010 E$="LIGNE N°"+STR$(L):YE=2:GOSUB 39000:LOCATE 4,1
22020 FOR C=1 TO NC:PRINT"col.";C;"- ";:FOR J=1 TO NLG:PRINT V$(J,C);"/";:NEXT:L
INE INPUT" : ",V$(L,C)
22030 IF LEN(V$(L,1))<>1 THEN 22100
22040 IF V$(L,1)="t" THEN V$(L,1)="T"
```

```
22050 IF V$(L,1)="=" OR V$(L,1)="T" THEN C=NC:GOTO 22100
22060 IF V$(L,1)="E" OR V$(L,1)="e" THEN CLS:LOCATE 1,50:PRINT"(correction)":L=L
-1:GOTO 22010
22070 IF INSTR("Qq",V$(L,1))=0 THEN 22100 ELSE C=NC:FQ=1
22100 NEXT:RETURN
23000 'INSERTION LIGNE apres ligne M
23010 NL=NL+1:FOR K=NL-1 TO M+1 STEP -1
23020 FOR C=1 TO NC:V$(K+1,C)=V$(K,C):NEXT:NEXT
23030 FOR C=1 TO NC:V$(M+1,C)="":NEXT
23040 RETURN
24000 'SUPPRESSION LIGNE M
24010 FOR K=M TO NL-1
24020 FOR C=1 TO NC:V$(K,C)=V$(K+1,C):NEXT:NEXT
24030 FOR C=1 TO NC:V$(NL,C)="":NEXT:NL=NL-1
24040 RETURN
25000 'REFUS
25010 BEEP:LOCATE 18,29:LINE INPUT"IMPOSSIBLE ! tapez ENTER ",R$:RETURN
26000 'CORREC NATURE LEGENDE
26010 CLS:LOCATE 3,20:PRINT"La NATURE de cette Colonne";C;"est ";NC$(C);" : OK ?
":TEX$="ON":GOSUB 50000:IF K=1 THEN 26060
26020 LOCATE 10,29:LINE INPUT"NATURE (T ou 0....6) : ",N$:FCOR=1
26030 IF LEN(N$)<>1 OR INSTR("Tt0123456",N$)=0 THEN BEEP:GOTO 26020
26040 NC$(C)=N$:IF N$="t" THEN NC$(C)="T"
26060 RETURN
27000 'MENU IMPRIM.
27010 DATA "Impression normale","Centrage du tableau","Lignes et colonnes numéro
tées",Quitter
27020 CLS:RESTORE 27010:NOPT=4:TIM$="PREPAREZ L'IMPRIMANTE":GOSUB 49000
27030 ON K GOTO 27040,27050,27060,27070
27040 TD=1:FL=0:FQI=0:GOTO 27100
27050 TD=(CPL-LT)/2:FL=0:FQI=0:GOTO 27100
27060 TD=1:FL=1:FQI=0:GOTO 27100
27070 TD=1:FQI=1
27100 RETURN
39000 ' Cadre simple d'une chaine E$ positionnée en YE
39010 XE=(80-LEN(E$))/2-2
39020 LE=LEN(E$)
39030 LOCATE YE-1,XE:PRINT CHR$(218);STRING$(LE+2,196);CHR$(191)
39040 LOCATE YE,XE:PRINT CHR$(179);" ";E$;" ";CHR$(179)
39050 LOCATE YE+1,XE:PRINT CHR$(192);STRING$(LE+2,196);CHR$(217)
39060 RETURN
39200 ' Cadre double d'une chaine E$ positionnée en YE
39210 XE=(80-LEN(E$))/2-2
39220 LE=LEN(E$)
39230 LOCATE YE-1,XE:PRINT CHR$(201);STRING$(LE+2,205);CHR$(187)
39240 LOCATE YE,XE:PRINT CHR$(186);" ";E$;" ";CHR$(186)
39250 LOCATE YE+1,XE:PRINT CHR$(200);STRING$(LE+2,205);CHR$(188)
39260 RETURN
39700 ' Cadre double de NLI lignes largeur LC positionné en YC
39710 XE=(80-LC)/2
39720 LOCATE YC,XE:PRINT CHR$(201);STRING$(LC-2,205);CHR$(187)
39730 FOR NN=YC+1 TO YC+NLI-2
39740 LOCATE NN,XE:PRINT CHR$(186);SPC(LC-2);CHR$(186)
39750 NEXT
39760 LOCATE YC+NLI-1,XE:PRINT CHR$(200);STRING$(LC-2,205);CHR$(188)
39770 RETURN
49000 ' AFFMENUA = AFFICHAGE de MENU
49010 ' NOPT=Nombre d'Options:TIM$=TITRE du MENU
49020 LOCATE 2,(80-LEN(TIM$))/2:PRINT TIM$
49030 TEX$="":FOR J=1 TO NOPT
49040 READ OPT$:TEX$=TEX$+LEFT$(OPT$,1)
49050 LOCATE J*2+13-NOPT,27:PRINT LEFT$(OPT$,1);" - ";OPT$
```

```
49060 NEXT
49070 GOSUB 50000
49080 RETURN
50000 ' MENUA - REPONSE A UN MENU
50010 LM=LEN(TEX$)
50020 LOCATE 24,(70-LM*2)/2:PRINT"Réponse (";
50030 FOR I= 1 TO LM-1:PRINT MID$(TEX$,I,1);",";:NEXT
50040 PRINT RIGHT$(TEX$,1);")";
50050 R$="":WHILE R$="":R$=INPUT$(1):WEND
50060 UPP$=R$:GOSUB 51000:R$=UPP$
50070 K=INSTR(TEX$,R$)
50080 IF K=0 THEN BEEP:GOTO 50050
50090 LOCATE 24,1:PRINT SPACE$(79);
50100 RETURN
51000 ' UPPERA - MISE EN MAJUSCULES
51010 FOR I=1 TO LEN(UPP$)
51020 IF ASC(MID$(UPP$,I,1))>96 AND ASC(MID$(UPP$,I,1))<123 THEN MID$(UPP$,I,1)=
CHR$(ASC(MID$(UPP$,I,1))-32)
51030 NEXT
51900 RETURN
64000 ' TRAITEMENT ERREUR DISK
64010 BEEP:ASTER$=STRING$(10,"*"):TAPENT$=ASTER$+" TAPEZ ENTER "+ASTER$
64015 IF ERL=10230 THEN RESUME 10200
64020 IF ERR=53 THEN PRINT ASTER$+" FICHIER NON TROUVE ";:GOTO 64300
64030 IF ERR=61 THEN PRINT ASTER$+" DISQUE PLEIN ";:GOTO 64300
64040 IF ERR=64 THEN PRINT ASTER$+" NOM DE FICHIER NON TROUVE ";:GOTO 64300
64050 IF ERR=70 THEN PRINT ASTER$+" DISQUETTE PROTEGEE ";:GOTO 64300
64060 IF ERR=71 THEN PRINT ASTER$+" DRIVE OUVERT OU VIDE ";:GOTO 64300
64070 IF ERR=72 THEN PRINT ASTER$+" ERREUR sur le DRIVE ";:GOTO 64300
64075 IF ERR=27 THEN PRINT ASTER$+" PLUS DE PAPIER ";:GOTO 64300
64080 IF ERR=24 OR ERR=25 OR ERR=57 THEN PRINT ASTER$+" IMPRIMANTE PAS PRETE ";:
GOTO 64300
64090 IF ERR=52 OR ERR=64 OR ERR=76 THEN PRINT ASTER$+" NOM DE FICHIER INCORRECT
";:GOTO 64300
64100 IF ERR=55 THEN CLOSE:RESUME 1000
64200 PRINT ASTER$;" ERREUR N°";ERR;"en Ligne";ERL;" ";ASTER$
64300 PRINT TAPENT$;:INPUT"",Q$:RESUME 1000
```

TARIFS postaux PTT (août 1987)

Poids (jusqu'à)	20 g	50 g	100 g	250 g	500 g	1000 g
LETTRE	2.20	3.70	5.60	12.30	15.30	20.00
PLI NON URGENT	2.00	2.70	3.70	7.40	10.90	15.40
PLI RECOMMANDE	15.90	17.40	19.30	26.00	29.00	33.70
PAQUET ORDINAIRE			3.70	7.40	10.90	15.40
PAQ. RECOMMANDE			10.40	14.10	17.60	22.10

T A B L I C

Edition automatique de Tableaux

(Michel Archambault - version février 89)

N - Nouveau tableau

C - Corrigé, Suite de saisie

I - Impression

S - Sauvegarde sur A:

R - Recharge un tableau en A:

Q - Quitter

Réponse (N,C,I,S,R,Q)

Suite de la page 44

```
655 IF ANNEE$="" THEN PRINT CHR$(7):LOCATE 10,43:PRINT "    ":GOTO 650
660 LOCATE 15,40:LINE INPUT CREDITACT$
665 IF CREDITACT$="" THEN PRINT CHR$(7):LOCATE 15,40:PRINT "         ":GOTO 660
670 CREDITACT=VAL(CREDITACT$)
680 LOCATE 12,38:LINE INPUT COMPTE$
690 IF COMPTE$="C" OR COMPTE$="c" THEN 720
700 IF COMPTE$="D" OR COMPTE$="d" THEN 720
710 PRINT CHR$(7):GOTO 680
720 COLOR 3,0
730 GOSUB 2680
740 OPEN "O",#1,FICHIER$
750 WRITE #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
760 CLOSE #1
765 SHELL DRIVER$+"ATTRIB +R "+FICHIER$
770 LOCATE 23,32:COLOR 0,2:PRINT " FICHIER CREE ":COLOR 3,0
780 FOR T=1 TO 2500:NEXT T
790 GOTO 410
800 '
810 ' ----- Mise a jour -----
820 '
830 CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,28:PRINT "M I S E   A   J O U R":COLOR 3,0
850 GOSUB 2620
860 LOCATE 16,9:PRINT "CREDIT ACTUEL :":LOCATE 16,35:PRINT "F"
870 LOCATE 16,49:PRINT "CREDIT :":LOCATE 16,68:PRINT "F"
880 LOCATE 18,49:PRINT "DEBIT  :":LOCATE 18,68:PRINT "F"
890 LOCATE 21,7:PRINT "VALEUR :"
900 LOCATE 23,12:COLOR 0,2:PRINT " F9   CREDIT           F10   DEBIT
  F7    FIN ":COLOR 3,0
910 LOCATE 21,57:PRINT "DATE :"
911 MS$=LEFT$(DATE$,2):MS=VAL(MS$)
912 ANNEE$=RIGHT$(DATE$,4)
913 IF MS=1 THEN MOIS$="JANVIER"
914 IF MS=2 THEN MOIS$="FEVRIER"
915 IF MS=3 THEN MOIS$="MARS"
916 IF MS=4 THEN MOIS$="AVRIL"
917 IF MS=5 THEN MOIS$="MAI"
918 IF MS=6 THEN MOIS$="JUIN"
919 IF MS=7 THEN MOIS$="JUILLET"
920 IF MS=8 THEN MOIS$="AOUT"
921 IF MS=9 THEN MOIS$="SEPTEMBRE"
922 IF MS=10 THEN MOIS$="OCTOBRE"
923 IF MS=11 THEN MOIS$="NOVEMBRE"
924 IF MS=12 THEN MOIS$="DECEMBRE"
930 COLOR 0,7
940 LOCATE 10,13:PRINT MOIS$
950 LOCATE 10,43:PRINT ANNEE$
960 LOCATE 12,38:LINE INPUT COMPTE$
970 IF COMPTE$="C" OR COMPTE$="c" THEN 1000
980 IF COMPTE$="D" OR COMPTE$="d" THEN 1000
990 PRINT CHR$(7):GOTO 960
1000 COLOR 3,0
1010 GOSUB 2680
1020 SHELL DRIVER$+"ATTRIB -R "+FICHIER$:OPEN "I",#1,FICHIER$
1040 OPEN "O",#2,"TEMPO"
1050 WHILE NOT EOF(1)
1060 INPUT #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
1070 WRITE #2,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
1080 WEND
1090 CLOSE #1
1100 KILL FICHIER$
1105 CREDIT=0:DEBIT=0
1110 LOCATE 16,25:COLOR 0,6:PRINT USING "######.##";CREDITACT:COLOR 3,0
1120 I$=INKEY$
1130 IF I$="C" THEN 1170
1140 IF I$="D" THEN 1250
1145 IF I$="F" THEN 1340
1150 GOTO 1110
1160 ' ----- Credit -----
1170 COLOR 0,7:LOCATE 16,58:LINE INPUT CREDIT$
1175 IF CREDIT$="" THEN 1220
1180 COLOR 3,0
1190 CREDIT=VAL(CREDIT$)
1200 CREDITACT=CREDITACT+CREDIT
1210 GOSUB 2700
1220 LOCATE 16,25:PRINT USING "######.##";CREDITACT
1225 LOCATE 16,58:COLOR 3,0:PRINT "         "
1230 GOTO 1110
1240 ' ----- Debit -----
1250 COLOR 0,7:LOCATE 18,58:LINE INPUT DEBIT$
1255 IF DEBIT$="" THEN 1300
1260 COLOR 3,0
1270 DEBIT=VAL(DEBIT$)
1280 CREDITACT=CREDITACT-DEBIT
1290 GOSUB 2700
1300 LOCATE 16,25:PRINT USING "######.##";CREDITACT
1305 LOCATE 18,58:COLOR 3,0:PRINT "         "
1310 GOTO 1110
1320 ' ----- Fin mise a jour -----
1340 CLOSE #2
1350 SHELL DRIVER$+"RENAME "+"TEMPO "+FICHIER$
1355 SHELL DRIVER$+"ATTRIB +R "+FICHIER$
1360 GOTO 410
1370 '
1380 ' ----- Suppression -----
1390 '
1400 '
1410 SUP=0:CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,30:PRINT "S U P P R E S S I O N":CO
LOR 3,0
1420 GOSUB 2620
1425 LOCATE 14,6:PRINT "N° ACTION A SUPPRIMER :"
1430 COLOR 0,7
1440 LOCATE 19,13:LINE INPUT MOIS$
1445 IF MOIS$="" THEN COLOR 3,0:GOTO 410
1450 IF LEN(MOIS$)<3 OR LEN(MOIS$)>9 THEN PRINT CHR$(7):LOCATE 10,13:PRINT "
     ":GOTO 1440
1460 LOCATE 10,43:LINE INPUT ANNEE$
1470 LOCATE 12,38:LINE INPUT COMPTE$
1480 IF COMPTE$="C" OR COMPTE$="c" THEN 1505
1490 IF COMPTE$="D" OR COMPTE$="d" THEN 1505
1500 PRINT CHR$(7):GOTO 1470
1505 LOCATE 14,29:LINE INPUT SUPP$
1506 IF SUPP$="" THEN PRINT CHR$(7):GOTO 1505
1509 SUP=VAL(SUP$)
1510 COLOR 3,0
1520 GOSUB 2680
1530 SHELL DRIVER$+"ATTRIB -R "+FICHIER$:OPEN "I",#1,FICHIER$
1540 OPEN "O",#2,"TEMPO"
1550 WHILE NOT EOF(1)
1560 INPUT #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
1565 IF NE=SUPP THEN 1575
1566 IF SUP=1 THEN NE=NE-1 ELSE 1570
1567 CREDITACT=CREDITACT+DEB-CRED
1570 WRITE #2,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
1573 GOTO 1650
1575 SUPP=-1:SOMME=CREDIT+DEBIT:CRED=CREDIT:DEB=DEBIT
1580 LOCATE 16,5:COLOR 0,7:PRINT USING "\      \";DAT$;:PRINT USING " #######.##
  F    ";SOMME;
1590 PRINT USING "\                                    \";VALEUR$:COLOR 3,0
1600 LOCATE 18,6:PRINT "A EFFACER (O/N) :"
1610 LOCATE 18,24:COLOR 0,7:LINE INPUT EFF$:COLOR 3,0
1620 IF EFF$="O" OR EFF$="o" THEN SUP=1:GOTO 1650
1630 IF EFF$="N" OR EFF$="n" THEN 1570
1640 PRINT CHR$(7):LOCATE 18,24:PRINT " ":GOTO 1610
1650 WEND
1740 CLOSE #1:CLOSE #2
1750 KILL FICHIER$
```

```
1760 SHELL DRIVER$+"RENAME "+"TEMPO "+FICHIER$
1770 SHELL DRIVER$+"ATTRIB +R "+FICHIER$
1780 GOTO 410
1790 '
1800 ' ----- Visualiser -----
1810 '
1820 Y=5:CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,27:PRINT "V I S U A L I S A T I O N":
COLOR 3,0
1830 GOSUB 2620
1840 COLOR 0,7
1850 LOCATE 10,13:LINE INPUT MOIS$
1855 IF MOIS$="" THEN COLOR 3,0:GOTO 410
1860 IF LEN(MOIS$)<3 OR LEN(MOIS$)>9 THEN PRINT CHR$(7):LOCATE 10,13:PRINT "
     ":GOTO 1850
1870 LOCATE 10,43:LINE INPUT ANNEE$
1880 LOCATE 12,38:LINE INPUT COMPTE$
1890 IF COMPTE$="C" OR COMPTE$="c" THEN 1920
1900 IF COMPTE$="D" OR COMPTE$="d" THEN 1920
1910 PRINT CHR$(7):GOTO 1880
1920 COLOR 3,0
1930 GOSUB 2680
1935 CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,27:PRINT "V I S U A L I S A T I O N":COLO
R 3,0
1940 OPEN "I",#1,FICHIER$
1950 WHILE NOT EOF(1)
1960 INPUT #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
1965 IF NE=0 THEN 2010
1970 SOMME=CREDIT+DEBIT
1975 IF DAT$=" "THEN 1981
1980 LOCATE Y,5:PRINT DAT$
1981 LOCATE Y,20:PRINT USING "######.##  F";SOMME
1982 IF VALEUR$=" " THEN 1991
1983 LOCATE Y,40:PRINT LEFT$(VALEUR$,30)
1990 LOCATE Y,75:PRINT USING "###";NE
1991 Y=Y+1:IF Y<21 THEN 2000
1992 Y=5:LOCATE 22,28:PRINT "* ESPACE POUR LA SUITE *"
1993 I$=INKEY$
1994 IF I$=" " THEN 1996
1995 GOTO 1993
1996 CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,27:PRINT "V I S U A L I S A T I O N":COLO
R 3,0
2000 SOMME=0
2010 WEND
2020 CLOSE #1
2025 LOCATE 22,28:PRINT "                          "
2030 LOCATE 22,34:COLOR 0,2:PRINT " FIN DE LISTE ":COLOR 3,0
2040 LOCATE 23,28:PRINT "* ESPACE POUR LA SUITE *"
2045 I$=INKEY$
2046 IF I$=" " THEN 410
2050 GOTO 2045
2060 '
2070 ' ----- Impression -----
2080 '
2090 CLS:GOSUB 2560:COLOR 2,0:LOCATE 2,30:PRINT "I M P R E S S I O N":COLOR 3,0
2100 GOSUB 2620
2110 COLOR 0,7
2120 LOCATE 10,13:LINE INPUT MOIS$
2125 IF MOIS$="" THEN COLOR 3,0:GOTO 410
2130 IF LEN(MOIS$)<3 OR LEN(MOIS$)>9 THEN PRINT CHR$(7):LOCATE 10,13:PRINT "
     ":GOTO 2120
2140 LOCATE 10,43:LINE INPUT ANNEE$
2150 LOCATE 12,38:LINE INPUT COMPTE$
2160 IF COMPTE$="C" OR COMPTE$="c" THEN 2190
2170 IF COMPTE$="D" OR COMPTE$="d" THEN 2190
2180 PRINT CHR$(7):GOTO 2150
2190 COLOR 3,0
2200 GOSUB 2680
2210 LOCATE 23,19:COLOR 0,2:PRINT " ALLUMEZ L'IMPRIMANTE ET APPUYEZ SUR ESPACE "
:COLOR 3,0
2220 I$=INKEY$
2230 IF I$=" " THEN 2240
2235 GOTO 2220
2240 IF COMPTE$="C" OR COMPTE$="c" THEN CMTP$="COURANT" ELSE CMTP$="DEPOT"
2245 LPRINT
2250 LPRINT SPC(30)"╔═══════════════════╗"
2260 LPRINT SPC(30)"│ RELEVE DE COMPTE  │"
2270 LPRINT SPC(30)"╚═══════════════════╝"
2278 LPRINT
2280 LPRINT "LE : ";DATE$;" \ COMPTE ";CMTP$
2290 LPRINT "═══════════════════════";:FOR I=1 TO LEN(CMTP$):LPRINT "=";:NEXT
I
2295 LPRINT CHR$(15):WIDTH "LPT1:",136
2300 LPRINT:LPRINT
2310 LPRINT "┌";:FOR I=1 TO 20:LPRINT "─";:NEXT 1:LPRINT "┬";:FOR I=1 TO 20:LPRI
NT "─";:NEXT I
2315 LPRINT "┬";:FOR I=1 TO 80:LPRINT "─";:NEXT I:LPRINT "┬";:FOR I=1 TO 10:LPRI
NT "─";:NEXT I:LPRINT "┐"
2320 LPRINT "│";:LPRINT SPC(7)"CREDIT";:LPRINT SPC(7)"│";:LPRINT SPC(7)" DEBIT";
:LPRINT SPC(7)"│";:LPRINT SPC(37)"VALEUR";:LPRINT SPC(37)"│";:LPRINT SPC(3)"DATE
";:LPRINT SPC(3)"│"
2330 LPRINT "├";:FOR I=1 TO 20:LPRINT "─";:NEXT I:LPRINT "┼";:FOR I=1 TO 20:LPRI
NT "─";:NEXT I
2340 LPRINT "┼";:FOR I=1 TO 80:LPRINT "─";:NEXT I:LPRINT "┼";:FOR I=1 TO 10:LPRI
NT "─";:NEXT I:LPRINT "┤"
2380 OPEN "I",#1,FICHIER$
2390 WHILE NOT EOF(1)
2400 INPUT #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
2405 IF NE=0 THEN 2450
2410 LPRINT "│";:LPRINT USING " ############.##  F ";CREDIT;
2420 LPRINT "│";:LPRINT USING " ############.##  F ";DEBIT;
2430 LPRINT "│";:LPRINT USING "\
                                \";VALEUR$;
2440 LPRINT "│ ";:LPRINT USING "\      \";DAT$;:LPRINT " │"
2450 WEND
2460 LPRINT "└";:FOR I=1 TO 20:LPRINT "─";:NEXT I:LPRINT "┴";:FOR I=1 TO 20:LPRI
NT "─";:NEXT I
2463 LPRINT "┴";:FOR I=1 TO 80:LPRINT "─";:NEXT I:LPRINT "┴";:FOR I=1 TO 10:LPRI
NT "─";:NEXT I:LPRINT "┘"
2465 LPRINT
2470 LPRINT SPC(51)"┌──────────────────────────┐"
2475 LPRINT SPC(51)"│ CREDIT ACTUEL │ ";:LPRINT USING "#######.##  F ";CREDITACT
;:LPRINT " │"
2478 LPRINT SPC(51)"└──────────────┴───────────────┘"
2480 CLOSE #1
2490 LPRINT CHR$(12)
2520 GOTO 410
2530 '
2540 ' ----- Sous-Programmes -----
2550 '
2560 COLOR 14,0
2563 FOR I=2 TO 78:LOCATE 23,I:PRINT "=";:NEXT I
2564 FOR I=2 TO 21:LOCATE 1,I:PRINT "=";:NEXT I
2565 FOR I=56 TO 78:LOCATE 1,I:PRINT "=";:NEXT I
2566 FOR I=2 TO 22:LOCATE I,1:PRINT "│":LOCATE I,79:PRINT "│":NEXT I
2567 LOCATE 1,1:PRINT "┌":LOCATE 1,79:PRINT "┐"
2568 LOCATE 23,1:PRINT "└":LOCATE 23,79:PRINT "┘"
2570 LOCATE 1,22:PRINT "═══════════════════════"
2580 LOCATE 2,22:PRINT "                       "
2590 LOCATE 3,22:PRINT "═══════════════════════"
2600 COLOR 3,0
2610 RETURN
2620 LOCATE 10,6:PRINT "MOIS :"
2630 LOCATE 10,35:PRINT "ANNEE :"
2640 LOCATE 10,50:PRINT "DATE (m/j/a) :"
2650 LOCATE 10,65:PRINT DATE$
2660 LOCATE 12,6:PRINT "COMPTE COURANT ou DEPOT (C/D) :"
```

```
2670 RETURN
2680 FICHIER$=LEFT$(MOIS$,1)+RIGHT$(MOIS$,1)+RIGHT$(ANNEE$,2)+COMPTE$
2690 RETURN
2700 COLOR 0,7:LOCATE 21,15:LINE INPUT VALEUR$
2710 LOCATE 21,64:LINE INPUT DAT$
2715 NE=NE+1
2720 WRITE #2,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
2722 CREDIT=0:DEBIT=0
2725 COLOR 3,0
2730 LOCATE 16,58:PRINT "      "
2740 LOCATE 21,15:PRINT "
2750 LOCATE 21,64:PRINT "         "
2770 RETURN
2780 IF ERR=57 AND ERL=2240 THEN 2790 ELSE 2840
2790 PRINT CHR$(7)
2800 LOCATE 22,24:COLOR 0,4:PRINT " L'IMPRIMANTE N'EST PAS ALLUMEE ":COLOR 3,0
2810 FOR T=1 TO 2500:NEXT T
2820 LOCATE 22,24:PRINT "                                  "
2830 RESUME 2220
2840 IF ERR=27 THEN 2850 ELSE 2900
2850 PRINT CHR$(7)
2860 LOCATE 22,23:COLOR 0,4:PRINT " PLUS DE PAPIER DANS L'IMPRIMANTE ":COLOR 3,0
2870 FOR T=1 TO 2500:NEXT T
2880 LOCATE 22,23:PRINT "                                  "
2890 RESUME
2900 IF ERR=53 THEN 2901 ELSE 3020
2901 IF ERL=1020 THEN GOSUB 2910 ELSE 2904
2902 RESUME 830
2904 IF ERL=1940 THEN GOSUB 2910 ELSE 2907
2905 RESUME 1820
2907 IF ERL=2370 THEN GOSUB 2910 ELSE 2909
2908 RESUME 2090
2909 IF ERL=3940 THEN RESUME 3660 ELSE 410
2910 PRINT CHR$(7)
2920 LOCATE 22,30:COLOR 0,4:PRINT " FICHIER NON TROUVE ":COLOR 3,0
2930 FOR T=1 TO 2500:NEXT T
2940 LOCATE 22,30:PRINT "                    "
2950 RETURN
3020 IF ERR=61 THEN 3030 ELSE 3130
3030 IF ERL=740 THEN GOSUB 3080 ELSE 3060
3040 RESUME 590
3060 IF ERL=1040 THEN GOSUB 3080 ELSE 410
3070 RESUME 830
3080 PRINT CHR$(7)
3090 LOCATE 22,31:COLOR 0,4:PRINT " DISQUETTE PLEINE ":COLOR 3,0
3100 FOR T=1 TO 2500:NEXT T
3110 LOCATE 22,31:PRINT "                    "
3120 RETURN
3130 IF ERR=64 THEN 3140 ELSE 3280
3140 IF ERL=740 THEN GOSUB 3230 ELSE 3170
3150 RESUME 590
3170 IF ERL=1020 THEN GOSUB 3230 ELSE 3200
3180 RESUME 830
3200 IF ERL=1940 THEN GOSUB 3230 ELSE 410
3210 RESUME
3230 PRINT CHR$(7)
3240 LOCATE 22,32:COLOR 0,4:PRINT " NOM INCORRECT ":COLOR 3,0
3250 FOR T=1 TO 2500:NEXT T
3260 LOCATE 22,32:PRINT "                    "
3270 RETURN
3280 IF ERR=70 THEN 3290 ELSE 3380
3290 IF ERL=740 THEN GOSUB 3330 ELSE 3310
3300 RESUME 590
3310 IF ERL=1040 THEN GOSUB 3330 ELSE 410
3320 RESUME 830
3330 PRINT CHR$(7)
3340 LOCATE 22,24:COLOR 0,4:PRINT " DISQUETTE PROTEGEE EN ECRITURE ":COLOR 3,0
3350 FOR T=1 TO 2500:NEXT T
3360 LOCATE 22,24:PRINT "                                  "
3370 RETURN
3380 IF ERR=71 THEN 3390 ELSE 3510
3390 IF ERL=740 THEN GOSUB 3470 ELSE 3410
3400 RESUME 590
3410 IF ERL=1020 THEN GOSUB 3470 ELSE 3430
3420 RESUME 830
3430 IF ERL=1940 THEN GOSUB 3470 ELSE 3450
3440 RESUME 1820
3450 IF ERL=2370 THEN GOSUB 3470 ELSE 410
3460 RESUME 2090
3470 PRINT CHR$(7)
3480 LOCATE 22,29:COLOR 0,4:PRINT " DISQUETTE PAS PRETE ":COLOR 3,0
3485 FOR T=1 TO 2500:NEXT T
3490 LOCATE 22,29:PRINT "                    "
3500 RETURN
3510 IF ERL=740 THEN 3520 ELSE 410
3520 PRINT CHR$(7)
3530 LOCATE 22,28:COLOR 0,4:PRINT " FICHIER DEJA EXISTANT ":COLOR 3,0
3540 FOR T=1 TO 2500:NEXT T
3550 LOCATE 22,28:PRINT "                    "
3560 RESUME 590
3570 '
3580 ' ----- Creation automatique nouveau mois -----
3590 '
3600 COMPTE$="C"
3610 TEST=0:FLAG=0
3620 MO$=LEFT$(DATE$,2):MO=VAL(MO$)
3630 JO$=MID$(DATE$,4,2):JO=VAL(JO$)
3640 AN$=RIGHT$(DATE$,4):AN=VAL(AN$)
3650 GOTO 3670
3660 LOCATE 15,22:COLOR 0,3:PRINT " CREATION AUTOMATIQUE NOUVEAU MOIS "
3661 LOCATE 17,30:COLOR 31,6:PRINT " Patientez S.V.P... ":COLOR 3,0
3663 MO=MO-1:IF MO=0 THEN 3665 ELSE 3670
3665 MO=12:AN=AN-1
3670 IF MO=1 THEN MOIS$="JANVIER"
3680 IF MO=2 THEN MOIS$="FEVRIER"
3690 IF MO=3 THEN MOIS$="MARS"
3700 IF MO=4 THEN MOIS$="AVRIL"
3710 IF MO=5 THEN MOIS$="MAI"
3720 IF MO=6 THEN MOIS$="JUIN"
3730 IF MO=7 THEN MOIS$="JUILLET"
3740 IF MO=8 THEN MOIS$="AOUT"
3750 IF MO=9 THEN MOIS$="SEPTEMBRE"
3760 IF MO=10 THEN MOIS$="OCTOBRE"
3770 IF MO=11 THEN MOIS$="NOVEMBRE"
3780 IF MO=12 THEN MOIS$="DECEMBRE"
3790 ANNEE$=STR$(AN)
3800 GOSUB 2680
3805 IF FLAG=0 THEN 3940
3810 IF TEST=1 THEN 3880
3820 OPEN "I",#1,FICHIER$
3830 WHILE NOT EOF(1)
3840 INPUT #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
3850 WEND
3860 CLOSE #1:CREDIT=0:DEBIT=0:VALEUR$="":DAT$="":NE=0
3870 MO=MO+1:IF MO=13 THEN 3875 ELSE 3878
3875 MO=1:AN=AN+1
3878 TEST=1:GOTO 3670
3880 OPEN "O",#1,FICHIER$
3900 WRITE #1,CREDITACT,CREDIT,DEBIT,VALEUR$,DAT$,NE
3910 CLOSE #1
3915 SHELL DRIVER$+"ATTRIB +R "+FICHIER$
3920 IF COMPTE$="D" THEN RETURN
3930 COMPTE$="D":GOTO 3610
3940 FLAG=1:OPEN "I",#1,FICHIER$:CLOSE #1:RETURN
```

INTERROGATIONS dBASE

Qui n'a pas au moins entendu parler de dBase ? Celui-ci est un programme de gestion de base de données, ceci veut dire de grandes possibilités d'interrogations.

Une base de données relationnelle est un programme permettant de gérer des fichiers en les organisant sous forme de table ou tableau possédant des informations de même nature dans chacune de ses colonnes ; chaque ligne de ce tableau peut être vue comme une fiche.
C'est aussi un outil permettant d'isoler ou retrouver rapidement des informations en spécifiant des critères de recherches selon le contenu d'une ou plusieurs colonnes : Dans un fichier de client "Qui a un compte débiteur ?", ou "Qui habite sur Paris ?", dans un stock "Combien reste-t-il de boulons de 12 ?", ...
Prenons un exemple pour explorer ensemble les possibilités d'interrogation qu'offre "dBase", et les produits qui lui sont compatibles.

LA CREATION DE NOTRE EXEMPLE.

Imaginons que nous soyons à la tête d'une bibliothèque, et que nous souhaitions gérer les mouvements de nos livres. Nous pouvons alors créer un fichier qui contiendra les références à chacun de nos livres (un thème, un auteur, un titre) ainsi que les dates de sortie et de retour de chaque livre. D'autre part une colonne ne pouvant contenir que les abréviations de "Oui" ou "Non" nous permettra de savoir si le livre est disponible ou s'il a été sorti de la bibliothèque.
En démarrant dBase, depuis le DOS avec la commande "DBASE", vous allez en général obtenir un menu vous permettant de commencer à travailler avec ce logiciel : Il s'agit du mode d'assistance. Celui-ci permet de commencer à travailler, mais limite les possibilités de dBase.
Pour échapper à ce mode d'assistance, frappez sur la touche "Escape" qui doit vous ramener à un mode de dialogue direct : le mode interactif. Votre écran doit s'effacer et vous devez voir apparaître un point vous signalant que dBase est à votre disposition pour exécuter n'importe quelle demande que vous pouvez lui formuler. Sachez qu'il y a moyen de configurer dBase pour qu'il démarre automatiquement avec ce mode de fonctionnement.
Le première chose à faire consiste à créer le fichier, c'est à dire déclarer à dBase la nature et le type du contenu de chacune de ces colonnes. Pour cela nous allons utiliser la commande "CREATE" suivi d'un nom permettant ultérieurement de retrouver ce fichier :

```
. CREATE LIVRES

     Nom champ    Type         Dim  Dec          Nom champ    Type         Dim  Dec
     ==========================================================================
  1  THEME        Caractère    10
  2  AUTEUR       Caractère    10
  3  TITRE        Caractère    28
  4  PAGES        Numérique     5    0
  5  DISPO        Logique       1
  6  DATE_PRET    Date          8
  7  DATE_RENDU   Date          8

CREATE          ||<D:>||LIVRES              ||Champ: 1/7          ||      ||   Min
```

La commande "CREATE" nous permet de remplir un tableau de la même nature que ci-dessus.
Un champ désigne le contenu d'une colonne sur une ligne, leurs noms permettent donc d'identifier chaque colonne. Pour une ligne, ou une fiche, donnée le champ "TITRE" permettra par exemple d'enregistrer "20000 LIEUES SOUS LES MERS", et le champ "THEME", "SF" comme science fiction.
A chaque champ est aussi associé un type d'information (par exemple : du texte, des nombres, des dates) et une dimension, c'est à dire un nombre maximum de caractères ou chiffres. Pour les champs numériques, il faut aussi préciser le nombre de chiffres après la virgule (le nombre de décimales).
Dans la structure, ci-dessus, de notre fichier "LIVRES" vous devez aussi remarquer un champ "DISPO" de type "Logique". Ceci signifie que ce champs ne pourra prendre que les valeurs "Vrai" ou "Faux". Si le livre est disponible, nous remplirons ce champ avec l'un des symboles suivants : "O" pour "Oui" ou "T" pour "True" (Vrai). Par contre si le livre n'est pas disponible, nous le remplirons avec "N" pour "Non" ou "F" pour "False" (Faux).
Pour terminer la description du fichier "LIVRES" vous appuyez en même temps sur les touches "Ctrl" et "End". "dBase" vous demande alors si vous souhaitez commencer à mettre des informations dans ce fichier ; vous pouvez répondre par l'affirmative en appuyant sur la lettre "O", et commencer à introduire les fiches du listing ci-dessous.
Rappelons ici quelques commandes de base qui permettent de mettre à jour un fichier :
. APPEND : Pour ajouter des enregistrements (et modifier les enregistrements existants), . EDIT : Permet de modifier les enregistrements existants, . BOWSE : Présente le fichier sous forme de table à l'écran, pour le visualiser ou le modifier.
Un fois l'une de ces commandes activée, les touches avec des flèches permettent de se déplacer de champ en champ, de fiche en fiche, et de modifier les informations.
Pour terminer le travail avec l'une de ces commandes vous appuyez soit sur les touches "Ctrl" et "Fin" pour valider les modifications, soit sur la touche "Esc" pour annuler la dernière modification. Vous vous trouvez alors avec l'indicatif de disponibilité de dBase, le point.
N'éteignez jamais votre ordinateur alors que vous venez de modifier un fichier (le disque serait dans un état lamentable), mais sortez d'abord de dBase avec la commande "QUIT". Celle-ci rendra la main au DOS après avoir enregistré toutes les informations.
Lorsque vous appellerez dBase de nouveau, utilisez la commande "USE LIVRES" pour signifier à dBase que vous voulez travailler de nouveau avec le fichier "LIVRES".

PREMIERE EXPLORATION DU FICHIER.

Les deux commandes de base pour visualiser le contenu d'un fichier sont "LIST" et "DISPLAY" (visualise).
"LIST" permet de faire défiler à l'écran tout le contenu du fichier, comme le montre le listing ci-après. "DISPLAY", quant à lui, si on ne lui précise rien de plus, ne montre qu'un enregistrement à la fois.
Lorsque "DISPLAY" est utilisé avec le paramètre "ALL", il va vous permettre lui aussi de visualiser tout le fichier, cependant "DISPLAY ALL" va faire une pause toutes les 20 lignes, et va attendre la frappe d'une touche. "LIST" par contre ne fait pas ces pauses, sur un fichier de plus de 20 lignes il ne reste à l'écran que les 20 dernières lignes.

En examinant ce listing, remarquez que le contenu de la colonne "DISPO" est soit ".F.", soit ".T.". A moins d'être en saisie/modification de fiche, dBase encadre, et demande d'encadrer, ces valeurs logiques par deux points.
Dans le cadre de notre exemple, vous allez rencontrer un problème lors de l'exécution de ces commandes : le listing fait plus de 80 colonnes de large et la présentation va souffrir de retours à la ligne inattendus.
Pour pallier à ceci, vous pouvez ne faire la liste que des colonnes qui vous intéressent en en précisant le nom :

```
. LIST THEME, AUTEUR, TITRE Enreg #
THEME      AUTEUR      TITRE
1  POESIE     B              LES FLEURS DU MAL
2  JEU        PA             APPRENEZ A MIEUX JOUER AU BR ...
```

```
. LIST Enreg #
 THEME   AUTEUR  TITRE                          PAGES DISPO DATE_PRET DATE_RENDU
1  POESIE     B       LES FLEURS DU MAL              185 .T.   20/02/89  28/02/89
2  JEU        PA      APPRENEZ A MIEUX JOUER AU BR   575 .T.   18/01/89  20/03/89
3  POLICIER   AC      10 PETITS NEGRES               230 .T.   16/03/89  09/03/89
4  POLICIER   LC      LE SAINT A MIAMI               215 .T.   20/03/89  15/03/89
5  POLICIER   LC      LE SAINT CONTRE M Z            190 .F.   15/03/89  12/03/89
6  JEU        CD JP   BRIDGE D'AUJOURD'HUI           400 .F.   10/03/89  09/03/89
7  POESIE     SE      LE PETIT PRINCE                248 .F.   21/03/89  19/03/89
8  BD         HERGE   TINTIN AU CONGO                 62 .F.   22/03/89  22/03/89
9  BD         HERGE   TINTIN EN AMERIQUE             62 .T.   21/03/89  21/03/89
10  JEU        BG LY  LE TREFLE BLEU (ANGLAIS)       170 .T.   20/02/89  22/03/89
11  JEU        RG     TRAITE DE JEU DE GO 1+2        520 .T.   11/02/89  14/03/89
12  POLICIER   ML     LES CONFIDENCES D'ARSENE LUP   290 .F.   18/03/89  07/03/89
13  POLICIER   ML     ARSENE LUPIN GENTLEMAN CAMBR   215 .T.   17/03/89  19/03/89
14  POLICIER   ML     ARSENE LUPIN CONTRE HERLOCK    320 .T.   14/03/89  19/03/89
15  SF         LRH    TERRE CHAMP DE BATAILLE 1+2   1000 .F.   22/03/89  21/03/89
16  SF         JV2    LE CHASCH                      215 .F.   17/03/89  15/03/89
17  SF         JV     DE LA TERRE A LA LUNE          450 .T.   22/03/89  17/03/89
18  SF         VV     LE MONDE DES NON AS            300 .F.   20/03/89  19/03/89
19  SF         JV     20000 LIEUES SOUS LES MERS     880 .T.   14/02/89  02/03/89
```

RECHERCHES DANS UN FICHIER.

Les commandes dBase qui permettent de parcourir un fichier, par exemple "LIST" ou "DISPLAY", supportent des compléments permettant de ne travailler qu'avec une partie du fichier.
L'un de ces complément est introduit par le mot "FOR" suivi d'une valeur logique. "FOR" permet de ne pre ndre en considération que les enregistrements pour lesquels cette v aleur log ique est vraie. Ainsi, si nous désirons la liste des livres disponibles :

```
. LIST THEME, AUTEUR,
TITRE FOR DISPO Enreg #  THEME      AUTEUR     TITRE
1  POESIE     B          LES FLEURS DU MAL
2  JEU        PA         APPRENEZ A MIEUX JOUER AU BR
3  POLICIER   AC         10 PETITS NEGRES
4  POLICIER   LC         LE SAINT A MIAMI
9  BD         HERGE      TINTIN EN AMERIQUE
10  JEU        BG LY      LE TREFLE BLEU (ANGLAIS)
11  JEU        RG         TRAITE DE JEU DE GO 1+2
13  POLICIER   ML         ARSENE LUPIN GENTLEMAN CAMBR
14  POLICIER   ML         ARSENE LUPIN CONTRE HERLOCK
17  SF         JV         DE LA TERRE A LA LUNE
19  SF         JV         20000 LIEUES SOUS LES MERS
20  BD         U+G        ASTERIX LEGIONNAIRE .
```

Tous les champs n'étant pas de type logique, dBase vous permet de construire des expressions logiques à partir de valeurs d'autre type. Vous avez à votre disposition les opérations suivantes :

< pour plus petit que, <= pour plus petit que ou égal à, > pour plus grand que, >= pour plus grand que ou égal à, # ou <> pour différent de.

Ces opérations permettent de faire la comparaison entre deux valeurs dBase de même type, quelque soit ce type.

Quelques explications supplémentaires sont nécessaires en ce qui concerne la comparaison de texte.

Un texte exprimé de manière littérale comme "SF" doit être encadré par des guillemets ou des crochets ("SF", 'SF' ou [SF]).
Si les textes ne sont pas de la même longueur, dBase ne compare que les premiers caractères (ceci est une option qui peut être modifiée). Ainsi les textes "APPRENEZ" et "APPR" sont considérés comme étant égaux.
Les majuscules et les minuscules sont différentes ; ainsi les textes "APPRENEZ" et "Apprenez" sont différents. Les majuscules sont classées avant les minuscules, nous avons : "A" < "Z" < "a" < "z". (Conclusion, "APPRENEZ" vient avant "Apprenez").

Le problème des accents

Il est parfois nécessaire de conserver les accents, en particulier sur les noms propres. Le poids ASCII des lettres accentuées est tel qu'il n'y a pas respect de l'ordre lexicographique et donc impossibilité d'effectuer un classement correct. Ainsi M onsieur Geté sera classé après Monsieur Gety car le poids ASCII du é est plus grand que y. Seul à notre connaissance Foxbase résout ce problème à l'aide du module "european.mem".

```
. LIST THEME,AUTEUR,TITRE FOR PAGES > 300 Enreg #
THEME      AUTEUR     TITRE
2  JEU        PA         APPRENEZ A MIEUX JOUER AU BR ...


. LIST THEME,AUTEUR,TITRE FOR THEME = "SF" Enreg #
THEME      AUTEUR     TITRE
15  SF         LRH        TERRE CHAMP DE BATAILLE 1+2 ... .
```

dBase permet aussi de combiner plusieurs résultats logiques grâce aux opérations ".AND.", ".OR." et ".NOT.". Le résultat de deux valeurs logiques combinées avec ".AND." sera vrai si l'une et l'autre est vraie ; celui de deux valeurs combinées avec ".OR." sera vrai si l'une ou l'autre est vraie. . LIST THEME, AUTEUR, TITRE FOR AUTEUR = "JV" .AND. PAGES <= 400 Enreg # THEME AUTEUR TITRE 16 SF JV2 LE CHASCH

```
. LIST THEME, AUTEUR, TITRE FOR AUTEUR = "JV" .OR. AUTEUR = "V
Enreg #
THEME      AUTEUR     TITRE
16  SF         JV2        LE CHASCH
17  SF         JV         DE LA TERRE A LA LUNE
18  SF         VV         LE MONDE DES NON AS
19  SF         JV         20000 LIEUES SOUS LES MERS
```

La première question demande la liste des ouvrages dont le code auteur est "JV" et dont le nombre de pages est inférieur à 400 ; la deuxième question demande la liste des ouvrages dont le code auteur commence par "JV" ou par "VV". Remarquez que cette deuxième question n'a pas été formulée "... FOR AUTEUR = "JV" .OR. "VV"". Ceci n'aurait rien voulu dire, car "AUTEUR = "JV"" forme bien un résultat logique, mais pas ""VV"". Il est donc impossible de combiner ces deux éléments par un ".OR.".

Nous pouvons élargir les possibilités de recherches dans la base de données en utilisant des fonctions dBase. Celle-ci sont des expressions dBase, possédant un mot clef, comme les commandes, et éventuellement des paramètres mis entre parenthèses. Ces parenthèses sont nécessaires, même si aucun paramètre n'est nécessaire à cette fonction.

Une première fonction digne d'intérêt est "UPPER (texte)". Celle-ci convertit en majuscule un texte qui peut contenir des minuscules. Ceci peut être utile pour atténuer l'effet de certaines erreurs de frappe.

Ainsi, "LIST ... FOR UPPER(TITRE) = "ARSENE LUPIN"" va permettre de retrouver les fiches dont le titre commence par "ARSENE LUPIN", même si celui-ci est écrit en minuscule.

Une autre fonction qui peut être intéressante est "DELETED()" ; celle- ci renvoie la valeur vraie si la fiche est marquée pour être effacée. "LIST FOR DELETED()" visualise toutes les fiches qui vont être détruites lors de la prochaine commande de compactage du fichier.

Le complément "FOR expression logique" peut aussi être mis à une commande telle que "EDIT". Seules seront alors accessibles en édition les fiches qui obéissent à la clause "FOR". Par exemple "EDIT FOR THEME = "SF"" vous permettra de ne modifier que les fiches des livres de science fiction.

Les commandes qui parcourent un fichier dBase, comme "LIST", "DISPLAY" et "EDIT" possèdent d'autres compléments que nous verrons ultérieurement permettant de voir d'une manière différente ce fichier. Ces commandes seront en fait beaucoup plus exploitables sur les fichiers triés.

RECHERCHE D'UN ENREGISTREMENT DANS UN FICHIER.

Le problème ici est différent, nous désirons obtenir le premier enregistrement obéissant à un critère donné, puis éventuellement les enregistrements suivants.

Nous avons deux possibilités pour ceci : conserver le fichier "en vrac", ou bien mettre en place des critères de tris qui vont permettre d'accélérer les recherches.

La méthode lourde qui peut donner satisfaction avec de petits fichiers en vrac, utilise deux commandes :

```
"LOCATE FOR ..." et "CONTINUE" : . LOCATE FOR AUTEUR = "ML" Enreg =
12 . DISPLAY TITRE Enreg #  TITRE
12  LES CONFIDENCES D'ARSENE LUP . CONTINUE Enreg =
13 . DISPLAY TITRE Enreg #  TITRE
13  ARSENE LUPIN GENTLEMAN CAMBR . CONTINUE Enreg =
14 . DISPLAY TITRE Enreg #  TITRE
14  ARSENE LUPIN CONTRE HERLOCK . CONTINUE Fin de sélection LOCATE. .
```

Le "LOCATE FOR ..." permet de trouver le premier enregistrement répondant au critère de recherche, les "CONTINUE" permettent de passer aux enregistrements suivants. Le fichier n'étant pas trié, les recherches sont exécutées en examinant toutes les fiches.

TRIS ET CRITERES DE RECHERCHE.

Avec dBase, nous pouvons trier physiquement un fichier sur disque, mais ce n'est pas la possibilité que nous allons prendre en considération.

Examinons la mise en place d'un critère de recherche qui nous permettra de considérer le fichier comme étant trié.

Le terme technique pour ce critère de recherche est "indexation". Une indexation consiste à créer un deuxième fichier beaucoup plus petit que le premier dans lequel les critères de tris (par exemple le nom de l'auteur) est trié, et à coté duquel se trouve le numéro de la fiche.

On peut faire une analogie avec la manière dont une bibliothèque est gérée. Sur les étagères se trouve les livres, qui peuvent être assimilés à notre fichier de données. A l'entrée de la bibliothèque se trouve une boîte à fiches où tout les livres sont enregistrés avec leur numéro d'étagère. Une recherche est beaucoup plus facile à faire dans cette boîte à fiches qu'au travers de toutes les allées de la bibliothèque.

Pour construire un index avec dBase, vous utilisez la commande "INDEX" :

```
. INDEX ON AUTEUR TO IND.AUT 100% indexé(s) .
21 Enregistrements indexé(s) . .
LIST THEME, AUTEUR, TITRE Enreg #  THEME      AUTEUR     TITRE
3  POLICIER   AC
10 PETITS NEGRES
1  POESIE     B          LES FLEURS DU MAL
```

Comme le montre le début de listing ci-dessus, la création de l'index permet de considérer le fichier comme trié sur le critère "AUTEUR". Cependant, la première colonne, celle des numéros de fiches montre que ceux-ci ne se suivent pas, le fichier de données n'est donc pas trié physiquement.

Le complément "ON" de la commande "index" permet d'introduire le critère de tri, le complément "TO" indique le nom du fichier d'index. Ceci va nous permettre de construire plusieurs index pour un même fichier de données.

Le critère d'indexation peut faire appel à des fonctions dBase, ainsi pour laisser une liberté de présentation (majuscules ou minuscules) dans le contenu des

fiches, nous pouvons utiliser la commande "INDEX ON UPPER(AUTEUR) TO IND_AUT".
La recherche dans un fichier indexé peut alors se faire très rapidement par l'une des deux commandes "SEEK" ou "FIND". La différence entre ces deux commandes est assez subtile, et concerne principalement la personne qui programme sous dBase.

OPTIMISATION D'UNE RECHERCHE.

Supposons que nous désirions optimiser la commandes "LIST ... FOR AUTEUR = "JV"" alors que nous travaillons avec un gros fichier.
Comme nous venons de le voir, une commande "SEEK" ou "FIND" nous permet de nous positionner plus rapidement dans un fichier indexé. Il nous faut maintenant pouvoir lister les fiches suivantes.
La commande liste possède deux autres types de complément "WHILE expression logique", et "NEXT nombre". Le premier de ces compléments permet d'exécuter la commande "LIST" tant que l'expression considérée est vraie ; le deuxième permet d'examiner un nombre donné de fiches.
(LISTING 4).

```
. FIND "JV" . DISPLAY AUTEUR,TITRE,THEME Enreg #
AUTEUR     TITRE                          THEME
17  JV         DE LA TERRE A LA LUNE          SF . SKIP Enreg. No
19 .
DISPLAY AUTEUR,TITRE,THEME Enreg #
AUTEUR     TITRE                          THEME
19  JV         20000 LIEUES SOUS LES MERS   SF . SEEK "LR" .
DISPLAY AUTEUR,TITRE,THEME Enreg #
AUTEUR     TITRE                          THEME
15  LRH        TERRE CHAMP DE BATAILLE 1+2  SF .
```

TRAVAIL A PLUSIEURS INDEX.

Avec ce que nous venons de voir, il peut sembler très intéressant de mettre à jour plusieurs critères de recherche à chaque modification du fichier, bien qu'un seul puisse effectivement être utilisé à la fois.
Considérons que nous voulions attaquer notre fichier de données soit par le thème, soit par le code de l'auteur, ou les première lettres sur titre. Nous allons donc créer 3 fichiers d'index :

```
. USE LIVRES INDEX IND_AUT . INDEX ON THEME + AUTEUR TO IND_THEM 100% indexé(s).
21 Enregistrements indexé(s). . INDEX ON UPPER(LEFT(TITRE,10)) TO IND_TITR 100% indexé(s).
21 Enregistrements indexé(s). . SET INDEX TO IND_AUT, IND_THEM, IND_TITR .
 LIST AUTEUR,TITRE,THEME NEXT 3 Enreg #
AUTEUR     TITRE                          THEME
3  AC          10 PETITS NEGRES              POLICIER
1  B           LES FLEURS DU MAL             POESIE
10  BG LY      LE TREFLE BLEU (ANGLAIS)      JEU . SET ORDER TO 2 Index maître : IND_THEM.NDX . GO TOP
. LIST AUTEUR,TITRE,THEME NEXT 3 Enreg #
  AUTEUR     TITRE                          THEME
8  HERGE       TINTIN AU CONGO               BD
9  HERGE       TINTIN EN AMERIQUE            BD
20  U+G        ASTERIX LEGIONNAIRE           BD . SET ORDER TO 3 Index maître : IND_TITR.NDX . GO TOP
. LIST AUTEUR,TITRE,THEME NEXT 3 Enreg #
  AUTEUR     TITRE                          THEME
3  AC          10 PETITS NEGRES              POLICIER
19  JV         20000 LIEUES SOUS LES MERS    SF
2  PA          APPRENEZ A MIEUX JOUER AU BR JEU .
```

Commentons les opérations précédentes.

La première commande "INDEX ON THEME + AUTEUR ..." constitue un nouvel index à partir de la juxtaposition du thème et du code de l'auteur. "+" est une opération dBase qui peut être faite sur des chaînes de caractères, le résultat étant la mise bout à bout des deux

Dans cet exemple, vous devez remarquer l'apparition d'un nouvelle commande "SKIP". Celle-ci fonctionne que nous soyons indexés ou non, elle permet de passer à l'enregistrement suivant.
Lorsqu'un index est en cours d'utilisation, toute modification faite dans le fichier de données et répercutée dans l'index. Ainsi si lors d'une commande "EDIT", on modifie le code de l'auteur de la fiche 19, en "VJ", la fiche semblera se déplacer pour aller se ranger à la fin du fichier.
Pour que cette mise à jour se fasse, il faut penser à déclarer l'index à chaque nouvelle utilisation du fichier de données : "USE LIVRES INDEX IND_AUT", ou bien si le fichier est déjà actif, mais pas encore l'index : "SET INDEX TO IND_AUT".

LISTING 4

```
 . FIND "JV" . LIST AUTEUR, TITRE,THEME WHILE AUTEUR = "JV" Enreg #
AUTEUR     TITRE                          THEME
17  JV         DE LA TERRE A LA LUNE          SF
19  JV         20000 LIEUES SOUS LES MERS     SF
16  JV2        LE CHASCH                      SF .
FIND "JV" . LIST THEME, AUTEUR, TITRE NEXT 4 Enreg #
  THEME      AUTEUR     TITRE
17  SF         JV         DE LA TERRE A LA LUNE
19  SF         JV         20000 LIEUES SOUS LES MERS
16  SF         JV2        LE CHASCH
4  POLICIER   LC         LE SAINT A MIAMI .
```

chaînes. Cette opération s'appelle concaténation.
L'avantage d'une telle indexation est qu'à l'intérieur d'un même thème, le fichier apparaîtra trié par auteur.

Ainsi "SET ORDER TO 2" permet de considérer le fichier comme étant trié par thème et auteur, "SET ORDER TO 3", de le considérer trié par titre, et "SET ORDER TO 3" par auteur.

champ numérique pour les fiches concernées.

Voici les réponse à ces questions :

```
. USE LIVRES INDEX IND_ AUT, IND_THEM,IND_TITR . COUNT FOR AUTEUR = "JV "
2 Enregistrements compté(s). . SET ORDER TO 2 Index maître : IND_THEM.NDX . FIND "SF" . AVERAGE PAGES
WHILE THEME = "SF "        5 Enregistrements traité(s). PAGES   569 . FIND "BD " . SUM PAGES WHILE
THEME = "BD "        4 Enregistrements additionné(s). PAGES   220 .
```

Le deuxième index introduit une nouvelle fonction dBase "LEFT" qui permet d'isoler les caractères de gauche d'une chaîne de caractères. Ainsi "UPPER(LEFT(TITRE,10)" prend en compte les 10 caractères de gauche du titre et les met en majuscules.

Une fois tous nos index construits, nous demandons à dBase de les gérer par une version étendue de la commande "SET INDEX". Dans cette commande se trouve la liste des index à gérer. Le premier de ces index "IND_AUT" est alors l'index maître, et les recherches dans le fichier sont faites d'après le critère de tri spécifié lors de la création de cet index.
De la même manière, nous pouvons déclarer ces index lors d'une première utilisation d'un fichier avec "USE" :

```
. USE LIVRES INDEX IND_AUT,
IND_THEM, IND TITR
```

Nous pouvons aussi utiliser "SET ORDER TO 0" pour n'utiliser aucun critère de tris, et voir le fichier comme si aucun index n'était déclaré, bien que ceux-ci continuent d'être mis à jour.

QUELQUES STATISTIQUES SUR NOTRE FICHIER.

dBase possède des commandes permettant d'extraire des statistiques dans un fichier : "Combien ai-je de livres pour tel auteur ?", "Quel est la taille moyenne d'un livre de science fiction ?", ou encore "Combien de pages de bandes dessinées avons-nous ?". Voici quelques commandes permettant d'extraire ces informations :

Dans cet exemple, nous avons utilisé les deux possibilités "FOR" ou "WHILE". Nous avons 2 livres de Jules Verne, un livre de science fiction fait en moyenne 569 pages, et nous avons 220 pages de bandes dessinées.
Avant de conclure, signalons une dernière possibilité ; supposons que nous ayons plusieurs questions à poser, et que toutes ces questions possèdent des critères communs.
Par exemple, nous ne nous intéressons qu'aux livres disponibles, et pour ces livres nous désirons savoir : combien de Jules Verne, combien de ... Cela peut paraître fastidieux dans toutes les questions que nous posons de spécifier à chaque fois : "... FOR ... .AND. DISPO".
Nous pouvons alors demander à dBase de poser un filtre sur la base de données. Tous les enregistrements n'obéissant pas à la condition exprimée dans le filtre seront ignorés. La commande pour cela est "SET FILTER TO ..." :

```
. USE LIVRES INDEX IND_AUT,IND_THEM,IND_TITR . SET FILTER TO DISPO . LIST AUTEUR,TITRE,THEME Enreg #
  AUTEUR        TITRE                        THEME
 3  AC          10 PETITS NEGRES             POLICIER
 1  B           LES FLEURS DU MAL            POESIE
10  BG LY       LE TREFLE BLEU (ANGLAIS)     JEU
 9  HERGE       TINTIN EN AMERIQUE           BD
17  JV          DE LA TERRE A LA LUNE        SF
19  JV          20000 LIEUES SOUS LES MERS   SF
 4  LC          LE SAINT A MIAMI             POLICIER
13  ML          ARSENE LUPIN GENTLEMAN CAMBR POLICIER
14  ML          ARSENE LUPIN CONTRE HERLOCK  POLICIER
 2  PA          APPRENEZ A MIEUX JOUER AU BR JEU
11  RG          TRAITE DE JEU DE GO 1+2      JEU
20  U+G         ASTERIX LEGIONNAIRE          BD
. COUNT FOR THEME = "POLICIER"        4 Enregistrements compté(s). .
```

La commande "SET ORDER numéro" permet de passer d'un index à l'autre. Les autres index sont mis à jour lors des modifications d'une fiche, et la recherche est effectuée grâce à l'index dont le numéro est spécifié dans le "SET ORDER".

. COUNT (FOR / WHILE) : compte un nombre de fiche, . SUM Champ_numérique (FOR / WHILE) : Additionne le contenu de ce champ numérique pour les fiches considérées, . AVERAGE Champ_numérique (FOR / WHILE) : Calcule la moyenne de ce

Une fois l'exploitation de votre fichier faite, il ne vous reste plus qu'à frapper le mot "QUIT" pour quitter dBase, ou "USE ..." pour continuer à travailler avec un autre fichier, travail pour lequel nous vous souhaitons beaucoup de plaisir.

Jean-Paul Pruniaux

NOS LISTINGS VOUS INTERESSENT ?

POUR VOUS EVITER DE LES SAISIR,
COMPATIBLES PC MAGAZINE
VOUS PROPOSE UNE DISQUETTE
CONTENANT:

TOUS LES LISTINGS DE CE NUMERO

Remplissez ce coupon et envoyez- le à: Compatibles PC magazine, service disquettes, 5/7, rue de l'Amiral Courbet 94160 Saint-Mandé avec un chèque à l'ordre de Laser Presse de 59 F ttc pour la France ou 99 F ttc pour l'étranger.

Nom: Prénom:
Adresse:
Code postal: Ville: Pays:

Offre commune à Compatibles PC magazine n°24 et le Hors-série Programmation/Listings

AB CLUB VOUS OFFRE UNE DISQUETTE AVEC 6 LOGICIELS !

PC-Kick: L'organisateur de bureau résident en mémoire. Un grand must ! Quelle que soit votre application en cours vous accédez instantanément à: bloc-notes, agenda calendrier, calculatrice, répertoire, alarme, machine à écrire, table ASCII, liste des départements, DOS, contrôle d'imprimantes.

Digger: un superbe jeu d'arcade se déroulant sous terre, pour un ou deux joueurs. Vous devez récupérer les pièces d'or et les diamants, tout en évitant les monstres qui vous poursuivent dans les galeries !

Dosedit2: utilitaire génial, Dosedit conserve en mémoire les dernières commandes DOS entrées au clavier; il permet de les rappeler et de les modifier. Dosedit vous fera économiser des milliers de frappe de touches. Dosedit2 ajoute un glossaire d'abréviations.

Dinotes: pour fixer des commentaires dans vos répertoires, en face des fichiers importants aux appellations trop souvent obscures. Ce programme pourra être utilisé avec des disquettes arc hivées: réutilisées quelques mois plus tard, elles resteront explicites.

Calc: une superbe calculatrice résidente qui fonctionne en modes: décimal, hexadécimal, octal et binaire. Elle est idéale pour les développeurs et effectue les conversions entre ces 4 modes et sur 32 bits pour les calculs d'adresses.

Keysub: un petit programme de réaffectation dynamique des touches du clavier: vous donnez la touche à redéfinir, vous pressez la touche (ou la combinaison de touches) qui doit s'y rapporter: voilà c'est fait.

Questionnaire à retourner avant le 31 août 1989 minuit

OFFRE RESERVEE AUX 5000 PREMIERES REPONSES

Pour en bénéficier, il vous suffit de remplir intégralement le bon ci-dessous et de l'adresser avant le 31 août 1989 minuit à: AB club, disquette PC mag, 27, rue de Montévidéo, 75016 Paris.

Nom : ..... Prénom ..... Age .....
Adresse : .....
Code postal : ..... Ville ..... Tél. : .....

Etes-vous :
☐ Etudiant ☐ Enseignant ☐ Employé
☐ Commerçant/artisan ☐ Cadre/cadre sup. ☐ Prof. libérale
☐ Autre (précisez) : ..... Possédez-vous un PC ? ☐ Au bureau ☐ Chez vous

Programmez-vous ? ☐ oui ☐ non

Quel genre de logiciels avez-vous l'intention d'acheter et quel budget leur destinez-vous ? .....

☐ Applicatifs ..... F ☐ Langages ..... F
☐ Jeux ..... F ☐ Utilitaires ..... F

AGENDA PROGRAMME DE GESTION

AGENDA est un programme écrit en GWBASIC, de gestion de fichiers à accès direct permettant de gérer son agenda personnel. Il s'utilise de facon très simple à l'aide du clavier et de menus déroulants qui facilitent l'accès aux différentes options du programme.

Après lancement du programme, l'écran est divisé en 4 parties. La première en haut à gauche est utilisée pour l'affichage des menus. La seconde, à droite, est un simple rappel des touches utilisables en fonction des options. Cette aide reste affichée en permanence quel que soit l'option en cours. Toutefois un beep sonore empêche toute fausse manoeuvre au clavier.
La partie basse de l'écran contient 2 cadres. Le plus petit à droite est réservé à l'affichage des différents messages destinés à l'utilisateur (erreurs, confirmations, entrée d'informations nécessaires au programme). La partie la plus importante est celle consacrée au "travail" sur les fiches. C'est dans ce cadre qu'elles apparaîssent. La plupart des options des deux menus utilisent cette portion d'écran.

LES MENUS DEROULANTS :

- Dérouler un menu : pour ce faire, il faut appuyer sur la touche de tabulation, ce qui rend actif le titre du menu. Une fois celui-ci mis en évidence, la touche 'ENTER' valide le choix et fait apparaître le menu correspondant avec toutes ses options. On peut passer d'un menu à l'autre même s'il est déjà déroulé en appuyant sur la touche de tabulation.
- Sélectionner une option : en appuyant sur la barre d'espace, on fait se déplacer une barre grise sur les options. La touche 'ENTER' valide l'option choisie.

LES OPTIONS PAR MENU :

A) Menu "Fichier" :
1) Lire : lecture du fichier. Les touches : '+' fiche suivante, '-' fiche précédente, 'q' (ou Q) quitter l'option Note : Si on tente une lecture après la fin du fichier ou avant la première fiche, un message avec beep sonore apparaît dans le cadre bas-droit de l'écran.

2) Imprimer : impression complète du fichier L'imprimante doit être connectée

3) Statut : statut du fichier Après validation de cette option, un tableau apparaît montrant le nom de chaque champ, leur type et surtout leur longueur, ce qui peut se révéler bien utile avant la saisie de fiches, par exemple. Le nom du fichier et le nombre de fiches qu'il contient sont également précisés.

4) Effacer : efface le fichier AGENDA.DAT Cette option détruit le fichier contenu sur la disquette. Une confirmation est demandée.

5) Retour DOS : retour définitif au système d'exploitation MS-DOS. Cette option fait sortir du programme et fait revenir l'utilisateur sous MS-DOS après fermeture du fichier. Pour accéder au programme, il faut recharger le GWBasic.

6) Aller DOS : retour temporaire au système d'exploitation MS-DOS. Cette option abandonne momentanément le programme afin de permettre l'exécution d'une commande MS-DOS. On relance le programme en tapant la commande EXIT.

B) Menu "Fiche" :
1) Ajouter : ajoute une fiche à la fin du fichier. Les noms des champs apparaissent, il suffit de remplir la fiche en tapant les données puis en les validant avec 'ENTER'.
Le champ "Date :" est automatiquement affecté de la date courante. A la fin de la saisie de la fiche, une confirmation est demandée. En cas d'erreur, on peut recommencer la saisie. Le fait de taper la lettre q (ou Q) dans le champ "NOM :" permet de quitter la saisie en cours. La confirmation de la fiche saisie entraîne le retour aux menus, ce qui implique qu'il faille à nouveau sélectionner cette option pour saisir une nouvelle fiche.

2) Modifier : modifie les données d'une fiche. Après validation de cette option apparaît dans le cadre bas-droit de l'écran un message demandant si l'on désire accéder à la fiche à modifier par son numéro. Deux cas peuvent alors se présenter :
a) l'utilisateur connaît le numéro de la fiche à modifier, dans ce cas, il tape ledit numéro afin de faire apparaître la fiche.
b) l'utilisateur ne connaît pas le numéro de la fiche, il doit alors taper la valeur 0 comme numéro de fiche. La première apparaît alors à l'écran et les touches '+' et '-' permettent d'accéder à la fiche désirée comme pour une lecture normale. Lorsque la fiche à modifier est affichée à l'écran, il faut appuyer sur la touche 'm' (ou M). Une flèche apparaît alors en face du premier champ. Cette flèche peut être déplacée le long des champs (jusqu'au champ "Date :") avec la barre d'espace. Une fois que la flèche est correctement positionnée en face du champ à modifier, il faut appuyer sur la touche 'ENTER'. La donnée actuelle du champ s'efface de façon à permettre la saisie de la nouvelle donnée. La validation de cette nouvelle donnée entraîne l'apparition du message "Autre modification ? (O/N)" permettant si besoin une nouvelle modification de champ dans la fiche.
Remarque : on peut quitter l'option avant la modification en appuyant sur la touche 'q' (ou Q).

3) Effacer : efface les données d'une fiche. La procédure est identique à celle décrite dans l'option "Modifier" jusqu'à ce que la fiche à effacer soit affichée à l'écran. Ceci fait, l'appui sur la touche 'e' (ou E) entraîne l'effacement de la fiche. Par mesure de sécurité, une confirmation est demandée.

4) Imprimer : imprime les données d'une fiche. La procédure est identique à celle décrite dans l'option "Modifier" jusqu'à ce que la fiche à imprimer soit affichée à l'écran. Ceci fait, l'appui sur la touche 'i' (ou I) lance l'impression de la fiche.

5) Sélectionner : permet de sélectionner

des fiches en fonction d'un critère de choix. Après validation de cette option apparaît à l'écran les noms des champs ainsi qu'en face du premier d'entre eux une flèche déplaçable avec la barre d'espace. Cette flèche doit être amenée en face du champ servant de "filtre" à la sélection. Ceci fait, l'appui sur la touche 'ENTER' permet d'entrer une chaîne de caractères qui servira de filtre. Après validation, le programme va rechercher dans le fichier toutes les fiches correspondant à ce critère de choix. Il le fera savoir en affichant le nombre de fiches trouvées. Celles-ci pourront alors être lues avec la touche '+'.

6) Quitter prgm : quitte le programme et revient sous GWBasic. Cette option permet de quitter le programme (après confirmation) et fait revenir sous GWBasic après avoir fermé le fichier.

REMARQUES COMPLEMENTAIRES :

Toutes les options des deux menus se terminent par un retour aux menus. L'appui sur une touche autre que celles nécessaires à l'option en cours entraîne un beep sonore signalant l'erreur.

STRUCTURE DU PROGRAMME :

Le fonctionnement du programme est essentiellement axé sur l'utilisation de nonbreux sous-programmes afin de minimiser les opérations répétitives. Le nombre de lignes de programme écrites pour l'excécution des différentes options est ainsi réduit au minimum.

Lignes 100 à 210 :

Lignes 100 à 210 : mise dans le tableau CH$ des noms des 7 champs plus la constante "n° de fiche :"

Lignes 130 et 140 : mise dans le tableau LC de la longueur de chacun des 7 champs.

Ligne 150 : définition d'une fonction-chaîne permettant d'éliminer les blancs à droite des chaînes entrées comme contenu des champs dans le cas ou celles ci n'occupent pas toute la longueur du champ. Ceci permet de gagner de la place en mémoire et de faire la sélection des fiches (voir option Sélectionner du menu fiche.)

Lignes 160 à 180 : création du fichier sur disque et définition de la structure des enregistrements.Le FIELD de la ligne 180 définit un champ TOTAL$ d'une longueur égale à celle d'une fiche afin de permettre son effacement (voir option Effacer du menu Fiche.)

Lignes 190 à 210 : 2 tableaux sont déclarés (de taille variable suivant le nombre de fiches.) Le premier contiendra

```
10 REM ************************************************************************
20 REM *                                                                      *
30 REM *            AGENDA ' : Gestion de fichier a acces direct              *
40 REM *                       pour agenda personnel.                         *
50 REM *                                                                      *
60 REM *                            Auteur : HUET Didier - Janvier 1989       *
70 REM *                                                                      *
80 REM ************************************************************************
90 REM
100 DEFINT A-Z:OPTION BASE 1:KEY OFF:COLOR 15,0:CLS
110 RESTORE 120:FOR I=1 TO 8:READ CH$(I):NEXT I
120 DATA " NOM :"," Prenom :"," Adresse :"," C. postal :"," VILLE :"," Telephone :"," Date :"," N°
fiche :"
130 RESTORE 140:FOR I=1 TO 7:READ LC(I):NEXT I
140 DATA 20,15,30,5,20,16,10
150 DEF FN ST$(A$)=LEFT$(A$+SPACE$(2),INSTR(A$+SPACE$(2),SPACE$(2))-1)
160 OPEN "agenda.dat" AS #1 LEN=116:XX=0
170 FOR I=1 TO 7:FIELD #1,XX AS XX$,LC(I) AS VC$(I):XX=XX+LC(I):NEXT I
180 FIELD #1,116 AS TOTAL$
190 NF=LOF(1)/116
200 IF NF=0 THEN DIM C$(3),S(3):GOTO 250
210 DIM C$(NF),S(NF)
220 REM
230 REM ----------------------------------------
240 REM
250 LOCATE 23,1:PRINT CHR$(200):LOCATE 23,80:PRINT CHR$(188):LOCATE 16,50:PRINT CHR$(201)
260 FOR Y=2 TO 22:LOCATE Y,1:PRINT CHR$(186):LOCATE Y,80:PRINT CHR$(186):NEXT Y
270 FOR X=2 TO 79:LOCATE 23,X:PRINT CHR$(205):LOCATE 10,X:PRINT CHR$(205):NEXT X
280 LOCATE 10,1:PRINT CHR$(204):LOCATE 10,80:PRINT CHR$(185):LOCATE 10,32:PRINT CHR$(202):LOCATE
16,80:PRINT CHR$(185):LOCATE 23,50:PRINT CHR$(202)
290 FOR Y=2 TO 9:LOCATE Y,32:PRINT CHR$(186):NEXT Y
300 FOR Y=17 TO 22:LOCATE Y,50:PRINT CHR$(186):NEXT Y
310 FOR X=51 TO 79:LOCATE 16,X:PRINT CHR$(205):NEXT X
320 COLOR 0,15:LOCATE 1,1:PRINT STRING$(80,32)
330 LOCATE 1,4:PRINT "Fichier":LOCATE 1,18:PRINT "Fiche":LOCATE 1,53:PRINT "TOUCHES"
340 LOCATE 3,34:PRINT " + ":LOCATE 5,34:PRINT " - ":LOCATE 7,34:PRINT " Q ":LOCATE 3,56:PRINT " M
":LOCATE 5,56:PRINT " E ":LOCATE 7,56:PRINT " I "
350 COLOR 15,0:LOCATE 3,38:PRINT "Fiche suivante":LOCATE 5,38:PRINT "Fiche precedente":LOCATE
7,38:PRINT "Quitter option"
360 LOCATE 3,60:PRINT "Modifier la fiche":LOCATE 5,60:PRINT "Effacer la fiche":LOCATE 7,60:PRINT
"Imprimer la fiche"
370 REM
380 REM ************************* MENUS ******************************
390 REM
400 M$(1)="Fichier":M$(2)="Fiche":XM(1)=4:XM(2)=18
410 K=1
420 COLOR 15,0:LOCATE 1,XM(K):PRINT M$(K)
430 T$=INPUT$(1):IF T$=CHR$(13) THEN 470 ELSE IF ASC(T$)<>9 THEN BEEP:GOTO 430
440 COLOR 0,15:LOCATE 1,XM(K):PRINT M$(K)
450 K=K+1:IF K>2 THEN K=1
460 GOTO 420
470 ON K GOTO 490,620
480 REM
490 REM ******************** MENU FICHIER ********************
500 REM
510 O$(1)="Lire":O$(2)="Imprimer":O$(3)="Statut":O$(4)="Effacer":O$(5)="Retour DOS":O$(6)="Aller
DOS"
520 L=10:XO=3:GOSUB 2140
530 W=1:YO=3
540 COLOR 0,15:LOCATE YO,XO+1:PRINT O$(W)
550 T$=INPUT$(1):IF T$=CHR$(13) THEN 600 ELSE IF ASC(T$)=9 THEN GOSUB 2210:GOTO 440 ELSE IF T$<>"
" THEN BEEP:GOTO 550
560 COLOR 15,0:LOCATE YO,XO+1:PRINT O$(W)
570 W=W+1:IF W>6 THEN W=1
580 YO=YO+1:IF YO>8 THEN YO=3
590 GOTO 540
600 ON W GOTO 750,800,910,1020,1100,1170
610 REM
620 REM ******************** MENU FICHE ********************
630 REM
6                                                                    4                                                                    0
O$(1)="Ajouter":O$(2)="Modifier":O$(3)="Effacer":O$(4)="Imprimer":O$(5)="Selectionner":O$(6)="Quitter
prgm"
650 L=12:XO=17:GOSUB 2140
660 W=1:YO=3
670 COLOR 0,15:LOCATE YO,XO+1:PRINT O$(W)
680 T$=INPUT$(1):IF T$=CHR$(13) THEN 730 ELSE IF ASC(T$)=9 THEN GOSUB 2210:GOTO 440 ELSE IF T$<>"
" THEN BEEP:GOTO 680
690 COLOR 15,0:LOCATE YO,XO+1:PRINT O$(W)
700 W=W+1:IF W>6 THEN W=1
710 YO=YO+1:IF YO>8 THEN YO=3
720 GOTO 670
730 ON W GOTO 1250,1450,1610,1710,1800,2040
740 REM
750 REM ******************** LECTURE DU FICHIER *************************
760 REM
770 T=0:GOSUB 2590
780 GOTO 420
790 REM
800 REM ******************** IMPRIMER LE FICHIER ************************
810 REM
820 COLOR 15,0:LOCATE 18,52:PRINT "LE FICHIER EST EN":LOCATE 19,52:PRINT "COURS D'IMPRESSION…"
830 LF=LOF(1)/116
840 NF=1
850 GET #1,NF
860 GOSUB 2510
870 NF=NF+1
880 IF NF>LF THEN LOCATE 20,53:COLOR 31:PRINT "FIN DU FICHIER":COLOR 15,0:FOR P=1 TO 4500:NEXT
P:GOSUB 2460:GOSUB 2210:GOTO 420
890 GOTO 850
900 REM
910 REM ******************** STATUT DU FICHIER ************************
920 REM
930 COLOR 0,15:LOCATE 18,58:PRINT " NOM DU FICHIER :":COLOR 15,0:LOCATE 20,62:PRINT "AGENDA"
940 FOR Y=11 TO 22:LOCATE Y,15:PRINT CHR$(179):LOCATE Y,23:PRINT CHR$(179):LOCATE Y,34:PRINT
CHR$(179):NEXT Y
950 COLOR 7:LOCATE 12,5:PRINT "Champs":LOCATE 12,17:PRINT "Type":LOCATE 12,25:PRINT
"Longueur":COLOR 15
```

tous les contenus du champ de sélection choisi dans l'option Sélectionner.Le second contiendra les numéros des fiches trouvées correspondant au critère de sélection.

Lignes 250 à 370 :
Affichage de l'écran de travail.Les bordures sont faites avec les caractères graphiques pris dans le jeu de caractères graphiques du PC 1512 en mode texte.

Lignes 400 à 470 :
Affichage et sélection des titres des deux menus. la ligne 470 effectue le branchement correspondant au menu choisi.

Lignes 500 à 610 :
Affichage du menu Fichier et sélection d'une option.La ligne 470 effectue le branchement correspondant au menu choisi.

Lignes 630 à 740 :
Idem que ci-dessus pour le menu Fiche.La ligne 730 effectue le branchement.

Lignes 750 à 790 :
Branchement au sous-programme de lecture du fichier.

Lignes 800 à 900 :
Impression de toutes les fiches du fichier.Les fiches sont lues une par une (ligne 850) et imprimées grâce au sous-programme d'impression (ligne 860). Lorsque la fin du fichier est atteinte (ligne 880), il y a branchement aux deux sous-programmes d'effacement de la zone basse de l'écran (zone de travail et cadre) et d'effacement du menu suivi d'un retour aux menus.

Lignes 910 à 1010 :
Affichage du tableau présentant la structure du fichier.Les noms des champs, leur type et leur longueur sont affichés ainsi que le nombre de fiches (lignes 980 et 990). Après une pause, la zone basse de l'écran s'efface (grâce au sous-programme en 2260 et 2210) ainsi que le menu et l'on revient aux menus.

Lignes 1020 à 1090 :
Effacement du fichier. Après confirmation, la ligne 1060 ferme le fichier et l'efface de la disquette.
Lignes 1100 à1240 :
Retour au DOS. Après confirmation, la ligne 1140 ferme le fichier, la zone de travail s'efface puis le menu et l'instruction SYSTEM fait revenir sous DOS.

Lignes 1170 à 1240 :
Retour temporaire au DOS.Après

```
960 FOR J=14 TO 20:LOCATE J,2:PRINT CH$(J-13):LOCATE J,17:PRINT "Alpha":NEXT J
970 RESTORE 970:FOR F=1 TO 7:READ H:LOCATE F+13,28:PRINT H:NEXT F:DATA 20,15,30,5,20,16,10
980 LOCATE 12,36:PRINT "Le fichier contient":LOCATE 14,36:PRINT "actuellement     fiches."
990 LF=LOF(1)/116:COLOR 7:LOCATE 14,49:PRINT USING "###";LF:COLOR 15
1000 FOR P=1 TO 5000:NEXT P:GOSUB 2260:GOSUB 2210:GOSUB 2460:GOTO 420
1010 REM
1020 REM ******************** EFFACER LE FICHIER *************************
1030 REM
1040 COLOR 15,0:LOCATE 19,52:PRINT "EFFACEMENT CONFIRME ? (O/N)"
1050 T$=INPUT$(1)
1060 IF T$="o" OR T$="O" THEN CLOSE:KILL "agenda.dat":GOTO 1080
1070 IF T$<>"n" AND T$<>"N" THEN BEEP:GOTO 1050 ELSE GOSUB 2460:GOSUB 2210:GOTO 420
1080 LOCATE 19,52:COLOR 31:PRINT "LE FICHIER EST EFFACE        ":FOR P=1 TO 5000:NEXT P:COLOR
15:GOSUB 2460:GOSUB 2210:GOTO 420
1090 REM
1100 REM ******************** RETOUR AU DOS ******************************
1110 REM
1120 COLOR 15,0:LOCATE 19,55:PRINT "ARRET CONFIRME ? (O/N)"
1130 T$=INPUT$(1)
1140 IF T$="O" OR T$="o" THEN CLOSE:GOSUB 2460:GOSUB 2210:CLS:SYSTEM
1150 IF T$<>"n" AND T$<>"N" THEN BEEP:GOTO 1130 ELSE GOSUB 2460:GOSUB 2210:GOTO 420
1160 REM
1170 REM ******************** ALLER AU DOS ***************************
1180 REM
1190 COLOR 15,0:LOCATE 18,55:PRINT "CONFIRMATION ? (O/N)"
1200 LOCATE 20,55:PRINT "Taper EXIT pour revenir"
1210 T$=INPUT$(1)
1220 IF T$="O" OR T$="o" THEN CLOSE:GOSUB 2460:GOSUB 2210:CLS:SHELL:RUN
1230 IF T$<>"N" AND T$<>"n" THEN BEEP:GOTO 1210 ELSE GOSUB 2460:GOSUB 2210:GOTO 420
1240 REM
1250 REM ******************** AJOUTER UNE FICHE *************************
1260 REM
1270 NF=LOF(1)/116
1280 NF=NF+1
1290 GOSUB 2320
1300 COLOR 15,0:LOCATE 14,67:PRINT USING "###";NF
1310 LOCATE 12,19:LINE INPUT V$:IF V$="q" OR V$="Q" THEN NF=NF-1:GOTO 1420
1320 LSET VC$(1)=V$
1330 CC=12
1340 FOR I=2 TO 6
1350 CC=CC+2
1360 LOCATE CC,19:LINE INPUT V$:LSET VC$(I)=V$
1370 NEXT I
1380 LOCATE 12,64:PRINT DATE$:LSET VC$(7)=DATE$
1390 LOCATE 19,55:PRINT "Fiche correcte ? (O/N)"
1400 T$=INPUT$(1):IF T$="n" OR T$="N" THEN GOSUB 2260:GOSUB 2460:GOTO 1290
1410 IF T$="o" OR T$="O" THEN PUT #1,NF ELSE BEEP:GOTO 1400
1420 ERASE C$,S:DIM C$(NF),S(NF):NS=0
1430 GOSUB 2260:GOSUB 2460:GOSUB 2210:GOTO 420
1440 REM
1450 REM ******************** MODIFIER UNE FICHE ************************
1460 REM
1470 GOSUB 2920
1480 T1$="m":T2$="M"
1490 IF NUM=0 THEN TT=1:GOSUB 2590
1500 IF NUM<>0 THEN TT=1:NF=NUM:GOSUB 2620
1510 IF T$="q" OR T$="Q" THEN GOSUB 2260:GOSUB 2460:GOSUB 2210:GOTO 420
1520 GOSUB 2750
1530 IF W=7 THEN LOCATE 12,64:PRINT STRING$(15,32)::LOCATE 12,64:PRINT DATE$:LSET
VC$(7)=DATE$:GOTO 1570
1540 LOCATE YF(W),19:PRINT STRING$(31,32)
1550 LOCATE YF(W),19:LINE INPUT V$
1560 LSET VC$(W)=V$
1570 GOSUB 2460:COLOR 15,0:LOCATE 19,52:PRINT "Autre modification":LOCATE 20,52:PRINT "dans cette
fiche ? (O/N)"
1580 T$=INPUT$(1):IF T$="o" OR T$="O" THEN GOSUB 2460:LOCATE YF(W),XF(W):PRINT "     ":GOSUB
2750:GOTO 1530
1590 IF T$="N" OR T$="n" THEN PUT #1,NF:GOSUB 2260:GOSUB 2460:GOSUB 2210:GOTO 420
1600 REM
1610 REM ******************** EFFACER UNE FICHE ************************
1620 REM
1630 GOSUB 2920
1640 T1$="e":T2$="E"
1650 IF NUM=0 THEN TT=3:GOSUB 2590
1660 IF NUM<>0 THEN TT=3:NF=NUM:GOSUB 2620
1670 IF T$="q" OR T$="Q" THEN 1690
1680 LSET TOTAL$="":PUT #1,NF:GOSUB 2460:LOCATE 19,52:PRINT "Cette fiche est effacee…":FOR P=1 TO
3000:NEXT P
1690 GOSUB 2260:GOSUB 2460:GOSUB 2210:GOTO 420
1700 REM
1710 REM ******************** IMPRIMER UNE FICHE ************************
1720 REM
1730 GOSUB 2920
1740 T1$="i":T2$="I"
1750 IF NUM=0 THEN TT=2:GOSUB 2590:IF T$="q" OR T$="Q" THEN 420 ELSE 1770
1760 IF NUM<>0 THEN TT=2:NF=NUM:GOSUB 2620:IF T$="q" OR T$="Q" THEN 420
1770 GOSUB 2510
1780 GOSUB 2260:GOSUB 2210:GOSUB 2460:GOTO 420
1790 REM
1800 REM ******************** SELECTION DE FICHES ***********************
1810 REM
1820 GOSUB 2320:GOSUB 2750
1830 LF=LOF(1)/116
1840 FOR F=1 TO LF:GET #1,F:C$(F)=FN ST$(VC$(W)):NEXT F
1850 IF W=7 THEN LOCATE 12,64:LINE INPUT W$:GOTO 1870
1860 LOCATE YF(W),19:LINE INPUT W$
1870 LOCATE 18,52:PRINT "Recherche des fiches avec":COLOR 7:LOCATE 19,52:PRINT W$:COLOR 15:LOCATE
20,52:PRINT "dans le champ ";:COLOR 7:PRINT CH$(W):COLOR 15
1880 NS=0
1890 FOR N=1 TO LF:IF INSTR(C$(N),W$)=0 THEN 1910
1900 NS=NS+1:S(NS)=N
1910 NEXT N
1920 LOCATE 22,55:COLOR 7:PRINT NS;:COLOR 15:PRINT "fiches trouvees."
1930 FOR B=1 TO NS:G=S(B):GET #1,G
1940 GOSUB 2870
1950 COLOR 15,0:LOCATE 14,67:PRINT USING "###";G
1960 LOCATE 12,64:PRINT VC$(7)
```

confirmation, la ligne 1220 ferme le fichier, la zone de travail s'efface puis le menu et l'instruction SHELL fait revenir temporairement au DOS. L'instruction RUN permet de relancer le programme après avoir tapé sous DOS la commande EXIT.

Lignes 1260 à 1430 :
Option Ajouter du menu Fiche.
Lignes 1270 et 1280 : incrémentation du nombre de fiches.
Ligne 1290 : branchement au sous-programme d'affichage des noms des champs dans la zone de travail.
Lignes 1300 à 1380 : saisie de la fiche.
Lignes 1390 à 1410 : confirmation de la saisie et écriture de la fiche dans le fichier.
Lignes 1420 : effacement des tableaux servant à la sélection de fiches et redimensionnement avec nouveau total de fiches. Mise à zéro du nombre de fiches sélectionnées.
Ligne 1430 : branchement aux divers sous-programmes d'effacements.
Lignes 1450 à 1600 :
Option Modifier du menu Fiche.
Ligne 1470 :
Branchement au sous-programme d'accès à une fiche.
Lignes 1490 et 1500 :
Branchement au sous-programme de lecture du fichier suivant numéro de fiche à modifier.
Lignes 1530 à 1560 :
Modification de la fiche.
Lignes 1570 à 1590 :
Autre modification éventuelle.
Lignes 1610 à 1690 :
Option Effacer du menu Fiche
Ligne 1630 :
Branchement au sous-programme d'accès à une fiche. Lignes 1650 et 1660 : branchement au sous-programme de lecture du fichier suivant numéro de fiche à effacer.
Ligne 1680 :
Effacement de la fiche. L'effacement proprement dit n'étant pas possible, la fiche est "mise à blanc" en la réécrivant avec le champ TOTAL$ mis à blanc (chaîne vide).

Ligne 1710 à 1780 :
Option Imprimer du menu Fiche.
Ligne 1730 : branchement au sous-programme d'accès à une fiche.
Lignes 1750 et 1760 : branchement au sous-programme de lecture du fichier suivant numéro de fiche à imprimer.
Ligne 1770 : branchement au sous-programme d'impression.

Lignes 1800 à 2030 :
Option Sélectionner du menu Fiche.

```
1970 U=1:FOR N=12 TO 22 STEP 2:LOCATE N,19:PRINT VC$(U):U=U+1:NEXT N
1980 T$=INPUT$(1)
1990 IF T$="+" THEN 2000 ELSE IF T$="q" OR T$="Q" THEN 2020 ELSE BEEP:GOTO 1980
2000 NEXT B
2010 FOR P=1 TO 4500:NEXT P
2020 GOSUB 2260:GOSUB 2460:GOSUB 2210:GOTO 420
2030 REM
2040 REM ******************** QUITTER PROGRAMME **********************
2050 REM
2060 COLOR 15,0:LOCATE 19,55:PRINT "ARRET CONFIRME ? (O/N)"
2070 T$=INPUT$(1)
2080 IF T$="o" OR T$="O" THEN CLOSE:GOSUB 2460:GOSUB 2210:CLS:END
2090 IF T$<>"n" AND T$<>"N" THEN BEEP:GOTO 2070 ELSE GOSUB 2460:GOSUB 2210:GOTO 420
2100 REM ********************************************************************
2110 REM
2120 REM ---------- SOUS-PROGRAMMES ----------
2130 REM
2140 REM ---------- s.p. deroulement d'un menu ----------
2150 REM
2160 LOCATE 2,XO:PRINT CHR$(201);:FOR C=1 TO L:PRINT CHR$(205);:NEXT C:PRINT CHR$(187)
2170 FOR Y=3 TO 8:LOCATE Y,XO:PRINT CHR$(186);O$(Y-2);STRING$(L-LEN(O$(Y-2)),32);CHR$(186):NEXT Y
2180 LOCATE 9,XO:PRINT CHR$(200);:FOR C=1 TO L:PRINT CHR$(205);:NEXT C:PRINT CHR$(188)
2190 RETURN
2200 REM
2210 REM ---------- s.p. effacement d'un menu ----------
2220 REM
2230 COLOR 0,0:FOR Y=9 TO 2 STEP -1:LOCATE Y,XO:PRINT STRING$(14,32):NEXT Y:COLOR 15,0
2240 RETURN
2250 REM
2260 REM ---------- s.p. effacement zone bas d'ecran ----------
2270 REM
2280 FOR Y=11 TO 22:LOCATE Y,2:PRINT STRING$(48,32):NEXT Y
2290 FOR Y=11 TO 15:LOCATE Y,50 :PRINT STRING$(30,32):NEXT Y
2300 RETURN
2310 REM
2320 REM ---------- s.p. affichage des champs ----------
2330 REM
2340 COLOR 0,15
2350 J=1:FOR Y=12 TO 22 STEP 2:LOCATE Y,6:PRINT CH$(J):J=J+1:NEXT Y
2360 LOCATE 12,55:PRINT CH$(7):LOCATE 14,55:PRINT CH$(8)
2370 RETURN
2380 REM
2390 REM ---------- s.p. erreur fichier ----------
2400 REM
2410 BEEP:COLOR 31,0:LOCATE 21,55:PRINT "LECTURE IMPOSSIBLE !"
2420 IF T$="+" THEN LOCATE 22,57:PRINT "Fin du fichier…" ELSE LOCATE 20,55:PRINT "Debut du
fichier…"
2430 FOR P=1 TO 5000:NEXT P:GOSUB 2460
2440 COLOR 15:RETURN
2450 REM
2460 REM ---------- s.p. effacement cadre bas d'ecran ----------
2470 REM
2480 COLOR 0,0:FOR Y=17 TO 22:LOCATE Y,51:PRINT STRING$(29,32):NEXT Y
2490 RETURN
2500 REM
2510 REM ---------- s.p. impression d'une fiche ----------
2520 REM
2530 NOM$=FN ST$(VC$(1))
2540 LPRINT NOM$;"  ";VC$(2)
2550 LPRINT VC$(3):LPRINT VC$(4);" ";VC$(5):LPRINT "Tel. : ";VC$(6)
2560 LPRINT
2570 RETURN
2580 REM
2590 REM ---------- s.p. lecture du fichier ----------
2600 REM
2610 NF=1
2620 GOSUB 2320
2630 GET #1,NF
2640 GOSUB 2870
2650 COLOR 15,0:LOCATE 14,67:PRINT USING "###";NF:LOCATE 12,64:PRINT VC$(7)
2660 U=1:FOR N=12 TO 22 STEP 2:LOCATE N,19:PRINT VC$(U):U=U+1:NEXT N
2670 T$=INPUT$(1)
2680 IF T$="+" THEN 2720
2690 IF T$="-" THEN 2730
2700 IF T$="q" OR T$="Q" THEN GOSUB 2460:GOSUB 2260:GOSUB 2210:RETURN
2710 IF T$=T1$ OR T$=T2$ THEN RETURN ELSE BEEP:GOTO 2670
2720 LF=LOF(1)/116:NF=NF+1:IF LF<NF THEN GOSUB 2390:NF=NF-1:GOTO 2670 ELSE 2630
2730 NF=NF-1:IF NF=0 THEN GOSUB 2390:NF=1:GOTO 2670 ELSE 2630
2740 REM
2750 REM ---------- s.p. deplacement fleche ----------
2760 RESTORE 2770:FOR M=1 TO 7:READ YF(M):NEXT M
2770 DATA 12,14,16,18,20,22,12
2780 FL$="->":FOR M=1 TO 6:XF(M)=3:NEXT M:XF(7)=52
2790 W=1
2800 COLOR 15,0:LOCATE YF(W),XF(W):PRINT FL$
2810 T$=INPUT$(1):IF T$=CHR$(13) THEN 2850 ELSE IF T$<>" " THEN BEEP:GOTO 2810
2820 LOCATE YF(W),XF(W):PRINT "  "
2830 W=W+1:IF W>7 THEN W=1
2840 GOTO 2800
2850 RETURN
2860 REM
2870 REM ---------- s.p. effacement donnees fiche ----------
2880 COLOR 0,0:FOR Y=12 TO 22 STEP 2:LOCATE Y,19:PRINT STRING$(31,32):NEXT Y
2890 LOCATE 12,64:PRINT STRING$(16,32):LOCATE 14,67:PRINT "  "
2900 RETURN
2910 REM
2920 REM ---------- s.p. demande No de fiche ----------
2930 REM
2940 COLOR 15,0:LOCATE 17,52:PRINT "ACCES FICHE PAR SON NUMERO :"
2950 LF=LOF(1)/116
2960 LOCATE 19,52:PRINT "Numero fiche (1 a";LF;") :";:LINE INPUT NUM$
2970 NUM=VAL(NUM$)
2980 IF NUM>LF THEN BEEP:GOSUB 2460:GOTO 2920
2990 IF NUM=0 THEN GOSUB 2460
3000 RETURN
```

Ligne 1820 : branchement au sous-programme d'affichage des champs et au sous programme de déplacement de la flèche de sélection de champ.
Ligne 1840 : lecture des fiches et remplissage du tableau C$ avec le contenu des champs de toutes les fiches correspondant au champ sélectionné diminue des blancs à droite.
Ligne 1860 : saisie de critères de sélection dans le champ choisi.
Ligne 1880 : mise à zéro du nombre de fiches trouvées.
Lignes 1890 à 1910 : recherche des fiches répondant au critère de sélection et incrémentation du nombre de fiches trouvées.Le tableau S est rempli avec le numéro des fiches sélectionnées.
Lignes 1930 à 2000 : lecture des fiches trouvées et affichage de ces dernières.
Lignes 2040 à 2090 : Option Quitter prgm du menu Fiche.
Ligne 2080 : après confirmation d'arrêt, fermeture du fichier et arret du programme, retour au GWBasic.

Lignes 2120 à 3000 :
SOUS-PROGRAMMES
Lignes 2140 à 2190 : s.p. de déroulement et affichage d'un menu.
Lignes 2210 à 2240 : s.p. d'effacement d'un menu.
Lignes 2260 à 2300 : s.p. d'effacement de la zone de travail du bas de l'écran.
Lignes 2320 à 2370 : s.p. d'affichage des champs dans la zone de travail.
Lignes 2390 à 2440 : s.p. d'affichage d'erreur de lecture du fichier (tentative de lecture avant la première fiche ou après la dernière).
Lignes 2460 à 2490 : s.p. d'effacement du cadre de la zone basse de l'écran.
Lignes 2510 à 2570 : s.p. d'impression d'une fiche. La chaîne contenue dans le champ "NOM :" (VC$(1)) est diminuée des blancs à droite pour que le prénom soit imprimé à bonne distance du nom.
Lignes 2590 à 2740 : s.p. de lecture du fichier.
Lignes 2750 à 2850 : s.p. de déplacement de la flèche de sélection de champ pour les options 'Modifier' et 'Sélectionner' du menu 'Fiche'.
Lignes 2870 à 2900 : s.p. d'affichage du contenu des fiches.
Lignes 2920 à 3000 : s.p. d'accès à une fiche par son numéro.

REMARQUES FINALES : A la fin de chaque option, il y a effacement de la zone de travail, du cadre de la zone basse de l'écran et du menu en cours grâce aux sous-programmes 2260, 2460 et 2210. Toutes les options se terminent par un retour aux menus.

```
Fichier        Fiche                                  TOUCHES
              Ajouter                 + Fiche suivante      M Modifier la fiche
              Modifier                - Fiche precedente    E Effacer la fiche
              Effacer                 Q Quitter option      I Imprimer la fiche
              Imprimer
              Selectionner
              Quitter prgm

--> NOM :        TERIEUR                          Date :  01-08-1989
    Prenom :     Alain                            N° fiche :   4
    Adresse :    Chemin Vert
    C. postal :  18000                            ACCES FICHE PAR SON NUMERO :
    VILLE :      BOURGES                          Numero fiche (1 a 10 ) 4
    Telephone :  48-58-48-58
```

```
Fichier        Fiche                                  TOUCHES
Lire                                  + Fiche suivante      M Modifier la fiche
Imprimer                              - Fiche precedente    E Effacer la fiche
Statut                                Q Quitter option      I Imprimer la fiche
Effacer
Retour DOS
Aller DOS

NOM :        DURAND                           Date :  01-08-1989
Prenom :     Marcel                           N° fiche :   2
Adresse :    13, Bd. Bleu
C. postal :  75008
VILLE :      PARIS
Telephone :  44-44-55-55
```

ANCIENS NUMÉROS
COMPLÉTEZ VOTRE COLLECTION...

N°1
L'éveil des compatibles : 15 PC à moins de 10 000 F L'explosion des logiciels juniors - Les mémoires de masse pour PC.

N°2
Impressions sur imprimantes - IBM New Look - Qui veut la tête de la 5"25 ? Turbo Basic - Les cartes graphiques Cartes de séjour pour programmes résidents.

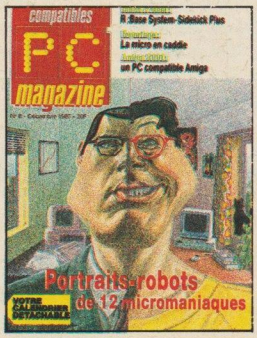
N°3
Imprimante Brother M-1709 -Amstrad PC 1512 disque dur à la loupe - Panorama des bases de données - Prix étudiés pour étudiants.

N°4
Tandon PC-X ou l'art de se démarquer - PS/2 8530 : compatible PC ou compatible IBM ? - Dix moniteurs sur la sellette - Test : êtes-vous doué pour la micro ?

N°5
Toshiba : deux portatifs - Diconix, imprimante à jet d'encre — Multitech, le compagnon mini AT - Modems à la carte - Sprint.

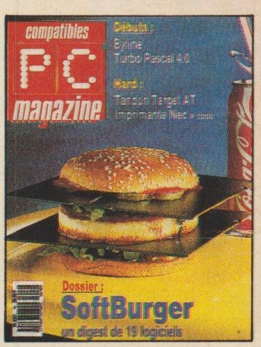
N°6
Carte 286 - Amstrad 1640 - TO16 de Thomson - PC1 de Commodore - Jasmin AT32 de Tran - Tulip PC2 - Vicki de Victor - Viva el C !

N°7
La galaxie dBase : les substituables et les complémentaires -- Sanyo 17+ : Baby 286 à petit prix -Carte Périscope - Quattro - Bourse : l'informatique en cause.

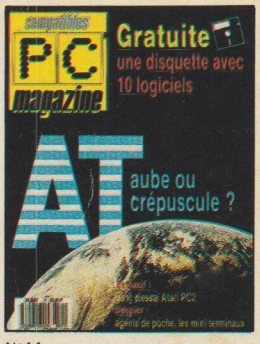


N°8
Bancs d'essai : R :Base System et Sidekick Plus - Amiga 2000 : un PC compatible amiga - La micro en caddie - Portraits-robots de 12 micro-maniaques.

N°9
Dossier PAO - Logiciels : Excel et RapidFile - Match : Thomson 16 XP-HD contre Amstrad PC 1640.

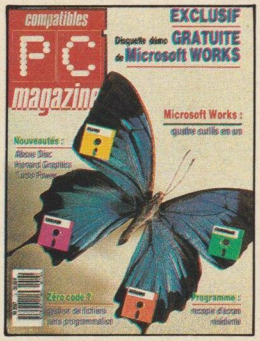


N°10
Tandon Target : un baby efficace -imprimante Nec P2200 - Knowledge Network - Byline - Turbo Pascal 4.0 - Softburger : un digest de 19 logiciels.

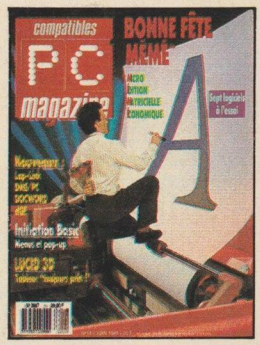


N°11
PC2 : Atari prend la carte du PC -AT, aube ou crépuscule ? —Agents de poche : les mini-terminaux - Windows 2.

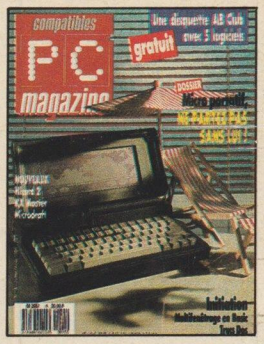
N°12
Test-match : Tandy 1000HX contre Olivetti PC1 - Pomper le minitel : un jeu d'adresses - Secrétariat high-tech : le futur au présent.



N°13
Microsoft Works : quatre outils en un - Zéro code : la gestion de fichiers sans programmation existe-t-elle ? Nouveautés : Above Disc, Harvard Graphics, Turbo Power.



N°14
Bonne fête Mémé : sept logiciels de micro édition à l'essai — Lucid 3D : tableur toujours prêt ! - Lap Link, DMS/PC, DocWord et dGE -Initiation Basic : menus et pop-ups.



N°15
Dossier : Micro portatif, ne partez pas sans lui ! Nouveaux : HiCard 2, KX Master et Microdraft - Initiation Basic : multifenêtrage - Trucs Dos.

Commandes d'anciens numéros de *Compatibles PC Magazine*

Veuillez m'adresser, pour 20 F par numéro :
☐n°1 ☐n°2 ☐n°3 ☐n°4 ☐n°5 ☐n°6 ☐n°7 ☐n°8 ☐n°9 ☐n°10 ☐n°11 ☐n°12 ☐n°13 ☐n°14 ☐n°15 ☐n°16 ☐n°17
Pour 25 F le numéro : ☐n°18 **Pour 22 F le numéro :** ☐n°19 ☐n°20 ☐n°21 ☐n°22 ☐n°23
☐la série des quinze numéros pour 225 F **☐la série de six numéros pour 100 F**

Nom : ..... Prénom : .....
Adresse : .....
Code postal ..... Ville ..... Téléphone : .....
Règlement par : ☐chèque bancaire ☐chèque postal ☐mandat
à adresser à :

Laser Presse, *Compatibles PC Magazine*, *Anciens numéros — 5/7, rue de l'Amiral Courbet 94160 Saint-Mandé.*